



Proyecto Final

Sistema de Estacionamiento

JETS

Elaborado por:

Sulem Martínez Aguilar
Daniel Mongeote Tlachy
Jesús Jacob Montiel Salas

05 de junio de 2023

Universidad Veracruzana

Contenido

Contenido	2
1. Introducción.....	3
Propósito.....	3
Alcance del producto	4
Alcance y limitaciones	4
Características Principales	4
Limitaciones.....	2
Supuestos.....	2
2. Descripción general.....	2
Modelo de proceso empleado: Cascada.....	2
Funciones del producto.....	4
Clases de usuario y características	4
Diagrama de Eriksson Penker y de BPMN	5
3. Requerimientos de interfaz externa.....	7
Interfaces de usuario	7
4. Características del sistema	15
Descripción	15
Requerimientos funcionales.....	15
5. Requerimientos no funcionales	17
Atributos de calidad del software	17
Disponibilidad:	17
Desempeño:	17
Mantenibilidad:.....	18
Seguridad e Integridad:	18
Reglas de negocio	18
Hechos	18
Restricciones	19
Habilitadores de acción	19
6. Planeación	20
Cronograma de actividades	20
7. Análisis de requerimientos	21
Diagramas de casos de uso.....	21

Descripción de los casos de uso.....	22
8. Diseño	33
Diagrama Entidad-Relación	33
Diagrama Relacional.....	34
.....	34
Diagrama de Clases	35
9. Implementación	36
Pruebas unitarias	36
10. Conclusión	46
11. Formatos Personal Software Process (PSP)	48
12. Bibliografía	48

1. Introducción

Propósito

Un software para la gestión de un estacionamiento surge de la necesidad de soluciones para la operación, gestión y control de estacionamiento, encargándose de las operaciones que involucran a un estacionamiento como sería la expedición de tarjetas para el uso de cajones, administración de costos, hacer pagos y cobros, las plumas de entrada y salida, comprobación de disponibilidad de lugares del estacionamiento, etc.

El software para la gestión de un estacionamiento posee distintos componentes, de los cuales cada uno se encarga de cumplir distintas tareas generando beneficios para los clientes y administradores del estacionamiento, permitiendo automatizar el proceso y, por lo tanto, el reducir el tiempo que consume el administrar y operar un estacionamiento, ahorrando una gran cantidad de tiempo, dinero y esfuerzo para todos los involucrados en el proceso. Este funcionará con la ayuda de un sistema de sensores, que estará instalado en todos los cajones del estacionamiento, y un sistema de validación de pago, que será externo al sistema y se ocupará cuando el cliente elija pagar con tarjeta.

Con dicho sistema se vuelve innecesaria la existencia de personas revisando cada nivel para comprobar la disponibilidad permitiendo al usuario una localización de espacio disponible más rápida, si en algún momento el estacionamiento llega a su cupo máximo este se encargara de negar el acceso al estacionamiento hasta que se desocupe un espacio, tiene un control de los tiempos de estancia de cada usuario lo que permite llevar un registro de la actividad del estacionamiento, se utilizará una

expedición de tarjetas para llevar el control de acceso de espacios, es decir, la disponibilidad de cajones.

En resumen, un software de gestión de un estacionamiento es una herramienta útil para mejorar la eficiencia y efectividad en la administración de un estacionamiento. Permite controlar el acceso, gestionar los pagos, gestionar la ocupación de lugares, y generar registros para controlar la administración de un estacionamiento.

Alcance del producto

Alcance y limitaciones

En esta sección se establecerá un alcance del Sistema de Estacionamiento JETS que incluirá la solución planeada, las características y las exclusiones de esta.

Características Principales

C1: Proporcionar Tarjeta

C2: Administración Registros y Tarjetas

C3: Administrar Costos

C4: Pago

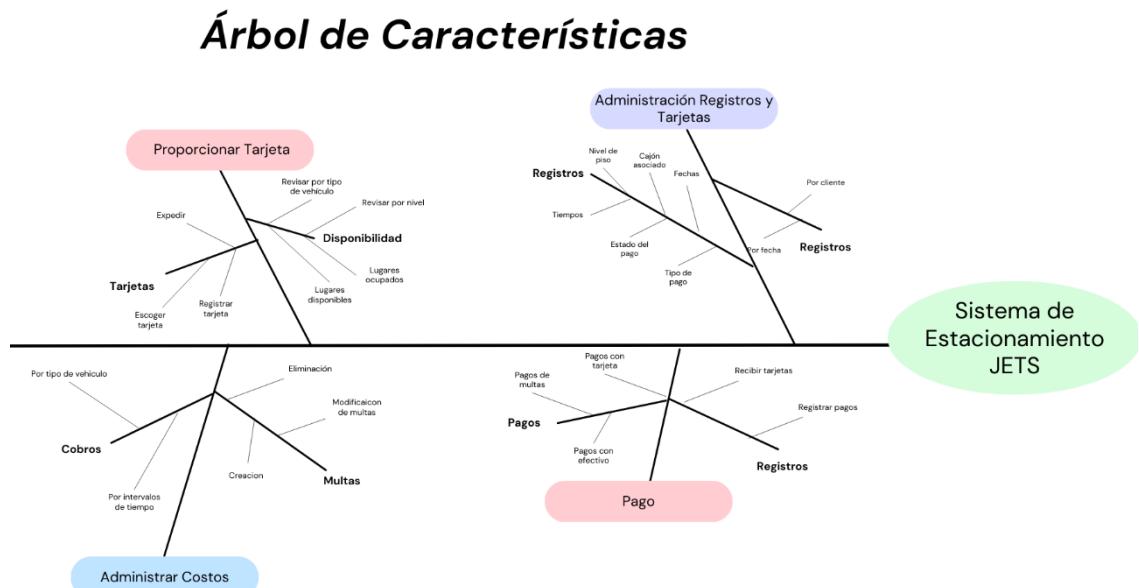


Ilustración 1. 1 Árbol de características para el Sistema de Gestión de un Estacionamiento

Característica	Versión del Alcance
Proporcionar Tarjeta	Monitoreo y control de disponibilidad de cajones por nivel, proporcionar tarjetas al cliente al entrar.
Administrar Registros y Tarjetas	Historial de las tarjetas por día, visualizar registros del estacionamiento, cambiar los estados de las tarjetas.
Administrar Costos	Modificar costos por vehículos, intervalos de tiempo. Cambiar costo de multas por pérdida de tarjeta, por equivocarse de cajón o por quedarse más tiempo de lo permitido.

Pagos	Recibir tarjetas de clientes al salir, hacer cobros al cliente por efectivo o tarjeta. Reportar tarjeta perdida y expedir multas.
-------	---

Ilustración 1. 2 Tabla de alcance

Limitaciones

- No se incluirá una tolerancia de tiempo para el cliente después de realizar pago debido a que el cliente pagará su tarjeta en el mismo lugar donde estará la salida.
- El sistema no contará un mecanismo para despachar tarjetas automáticamente.
- El estacionamiento no contará con la posibilidad de permitir el acceso al estacionamiento a vehículos de mayor escala, comparado con la de un automóvil particular.
- El sistema no proveerá soporte para plataformas móviles.
- El sistema no tendrá interacción directa con el cliente.
- El sistema no permitirá la comunicación por medio de mensajes entre usuarios.
- El sistema sólo admitirá pago por efectivo o por medio de un sistema de validación de pago. Sin embargo, este sistema de validación de pago será externo al sistema.

Supuestos

- El rendimiento y disponibilidad del sistema de gestión de estacionamiento depende parcialmente de los sistemas de sensores y de validación de pago.
- El sistema de los sensores siempre estará disponible.
- Todos los niveles del estacionamiento siempre se encuentran disponibles para uso, es decir, nunca tendrán que cerrarse por mantenimiento.
- El despachador no podrá entrar al sistema fuera del horario del establecimiento.
- Si el estacionamiento se llena por completo, se negará la entrada a cualquier vehículo.
- Si no se cuenta con la tarjeta de un cajón, el lugar no puede ser otorgado por el despachador hasta que el administrador imprima la tarjeta perdida y esta se registre en el sistema.
- El cliente siempre se irá al cajón asignado por el despachador.

2. Descripción general

Modelo de proceso empleado: Cascada

El modelo en cascada es un procedimiento lineal que se caracteriza por dividir los procesos de desarrollo de software en sucesivas fases del proyecto de manera escalonada. Estas fases se ejecutan solamente una vez y no se puede avanzar a la

siguiente fase hasta que se termine la que se está realizando en un determinado momento; de la misma manera, una vez que una fase se concluya y se empiece con la fase siguiente, no se puede regresar a hacer correcciones de la fase concluida.

El modelo de cascada se compone por 5 fases:

- **Comunicación:** Es la primera fase del proceso. Aquí se reúnen los todos los requisitos necesarios para el proyecto. Se deben recabar todos los posibles requerimientos para así generar un sistema con calidad. Cabe recalcar que los requerimientos deben ser congruentes y coherentes para apegarse a la realidad, para así evitar malentendidos y confusión a la hora de elaborar el proyecto.
- **Planeación:** En esta fase se establecen los tiempos que durará cada etapa del proyecto en base a la complejidad de cada fase; esto se hace con la finalidad de aprovechar al máximo el tiempo en que se empieza a desarrollar el proyecto hasta el *deadline*, además de evitar retrasos en la entrega.
- **Modelado:** Esta etapa consiste en el diseño del sistema en base a los requisitos obtenidos en la primera fase. Cabe aclarar que en esta fase no se hace ninguna implementación en código, más bien se establecen las especificaciones del sistema.
- **Construcción:** Esta es la fase donde se lleva a cabo la codificación del programa, donde se toma toda la información de la etapa anterior para implementarla en forma de código y así crear un producto funcional de calidad. A su vez, en esta etapa se hacen las pruebas correspondientes para verificar que el programa no tenga ningún fallo.
- **Despliegue:** Aquí finaliza el modelo, con el producto terminado y entregado para su lanzamiento.

Generalmente, el modelo en cascada es utilizado en proyectos cortos que están bien definidos, cuyos requisitos son estáticos.

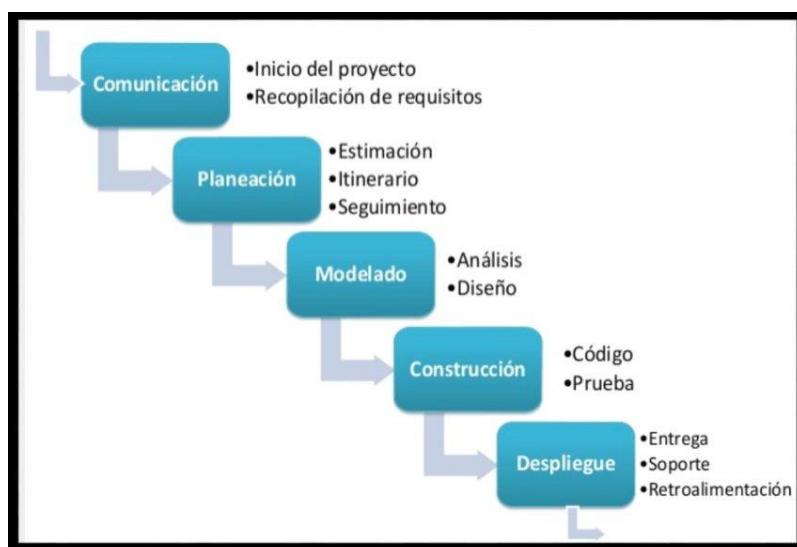


Ilustración 2.1 Modelo Cascada

Funciones del producto

Las funciones principales que el producto tendrá serán las siguientes:

- Permitirá llevar un registro de la disponibilidad de los cajones, para saber con más facilidad qué cajones están disponibles sin que haya personas revisando cada nivel.
- Desplegará en la pantalla la ubicación de la disponibilidad de espacios.
- Permitirá registrar la hora y fecha de entrada de un cliente automáticamente cuando se le proporcione una tarjeta a este.
- Permitirá registrar la hora y fecha de salida automáticamente después de que el despachador haya recibido la tarjeta, y haya validado el pago del cliente.
- Permitirá generar automáticamente el pago total cuando ingrese el identificador de la tarjeta para realizar el cobro.
- Permitirá indicar si se pudo realizar el pago, por efectivo o por tarjeta, o si el pago está pendiente.
- Permitirá asignar multas a los clientes.
- Dispondrá de un historial con todos los registros hechos que han sido registrados por día, el cual se puede consultar en cualquier momento.
- Permitirá registrar nuevas tarjetas en caso de que se lleguen a perder.
- Posibilitará cambiar algunos aspectos de cómo se calculan los costos del estacionamiento.
- Permitirá cambiar el estado del pago manualmente y de los cajones.
- Se podrá comunicar con el sistema de sensores para que se pueda actualizar automáticamente la disponibilidad de los cajones.
- Permitirá generar y descargar reportes con base al historial de todas las tarjetas registradas.
- Permitirá registrar usuarios de tipo administrador o de tipo despachador.

Clases de usuario y características

Usuarios favorecidos y directos:

- **Despachador:** Es el encargado de proporcionar las tarjetas del estacionamiento. Se encarga de revisar la disponibilidad de todos los cajones en el estacionamiento e informar de esta al cliente para que seleccione el respectivo nivel donde le asignara una tarjeta que corresponde a un cajón.
- **Despachador (Validador):** Se encarga de generar el cobro y de atender el pago de estacionamiento del Cliente dependiendo de ciertos parámetros (tiempo, cajón, multa) y de levantar la pluma en caso de que el pago sea aceptado.
- **Administrador:** Es el encargado de la gestión del sistema. Crea las tarifas de los horarios de cobro del estacionamiento y se encarga de generar sus respectivos reportes (visualizar los detalles de los registros durante toda la

jornada laboral) y de registrar nuevas tarjetas para el caso en que el Despachador solicite un nuevo lote.

Usuarios favorecidos e indirectos:

- **Cliente:** Son los individuos que interactúan con el despachador, están encargados de seleccionar un nivel del estacionamiento (dependiendo del tipo de automóvil que estén usando) para que así aseguren su inmobiliario dependiendo del tiempo que estén usando y de pagarle al despachador dependiendo del tiempo en que se queda.

Diagrama de Eriksson Penker y de BPMN

El diagrama de Eriksson Penker del modelo de negocio es:

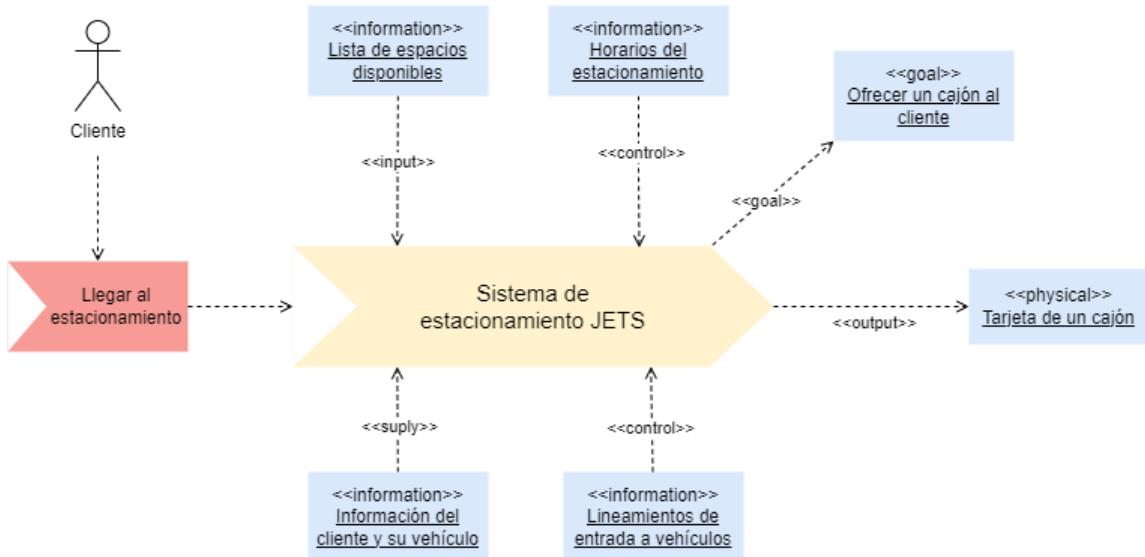


Ilustración 2.2.- Diagrama Eriksson Penker del SGE

El diagrama de BPMN del modelo de negocio es:

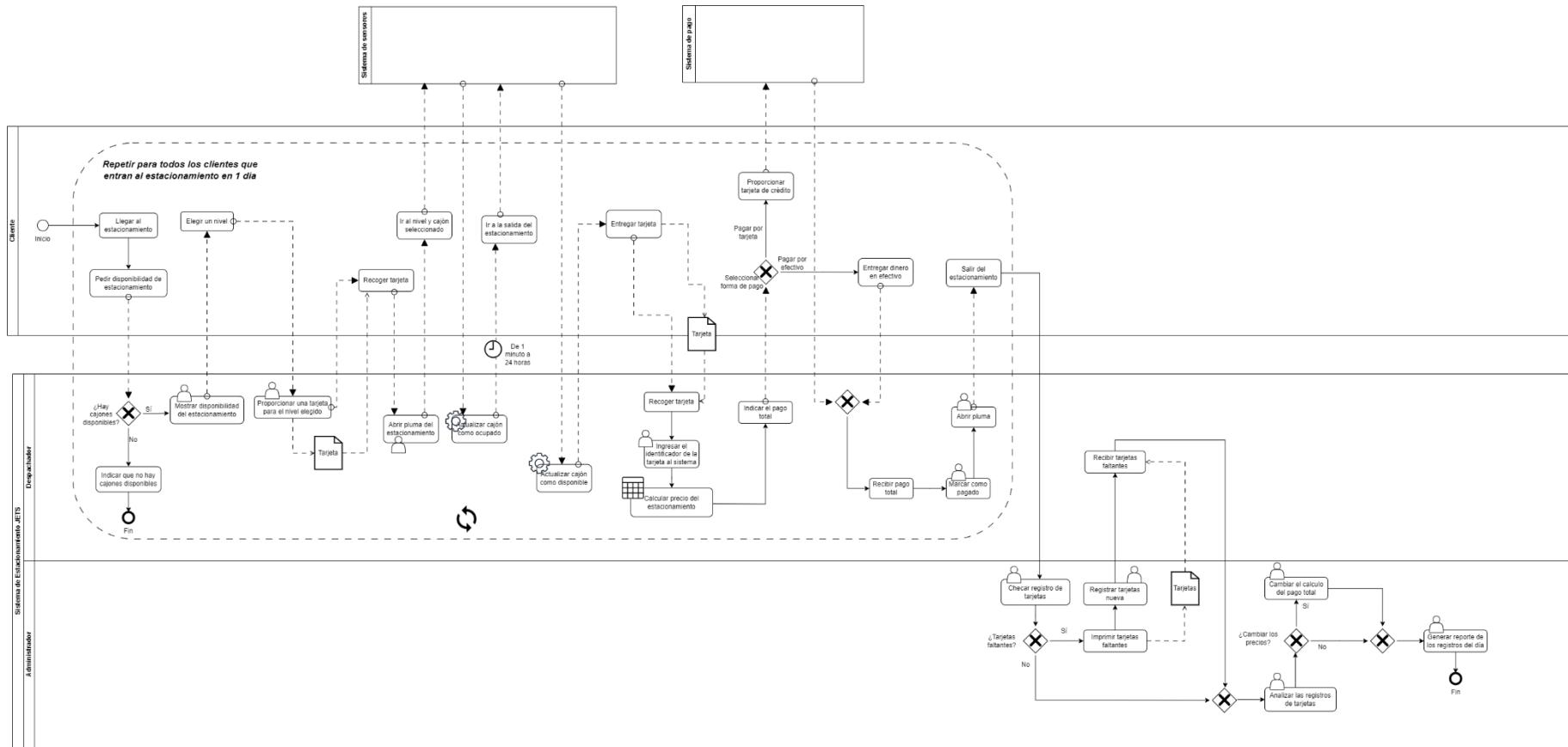


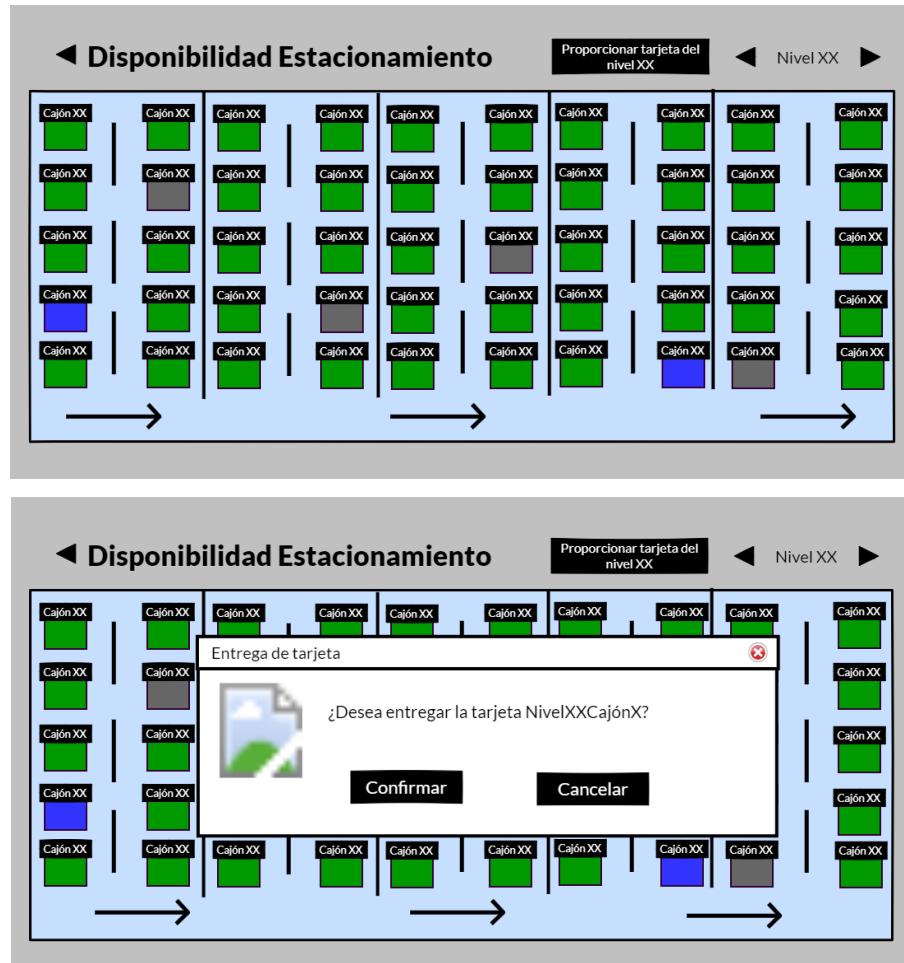
Ilustración 2.3.- Diagrama BPMN del Sistema de gestión de estacionamiento

3. Requerimientos de interfaz externa

Interfaces de usuario

Los prototipos de las pantallas del sistema son las siguientes:

CU01 – Proporcionar tarjeta



CU02 – Cobrar tarjeta

The image shows a user interface for 'Cobrar tarjeta'. It features a header with a back arrow and the title 'Cobrar tarjeta'. Below this is a section labeled 'Cobrar' with a button. A form area contains the label 'Identificador de tarjeta:' followed by an input field ('Ingresar el identificador de tarjeta') and a 'Mostrar' button. The rest of the page is a large, empty white space.

Cobrar tarjeta

Cobrar

Identificador de tarjeta: Mostrar

Código : XXXXX
Nivel: XX
Cajón: XX
Tipo de vehículo: XXXXX
Estatus de la tarifa: XXXXX
Fecha de Entrada: XX-XX-XXXX
Hora de Entrada: XX:XX:XX
Nombre del Despachador: XXXX XXXX XXXX

Iniciar Proceso de Pago

Cobrar tarjeta

Cobrar

Identificador de tarjeta: Mostrar

Fecha de Salida: XX-XX-XXXX Hora Actual: XX
Hora de Salida: XX:XX:XX
Total de minutos: XXX

Precio Total: XXX MXN

Asignar Multa(s) ▾

Asignar Método de Pago ▾

Confirmar Pago

Cobrar tarjeta

Cobrar

Identificador de tarjeta: Mostrar

Fecha de Salida: XX-XX-XXXX Hora Actual: XX
Hora de Salida: XX:XX:XX
Total de minutos: XXX

Precio Total: XXX MXN

Asignar Multa ▾

- Ninguna
- Por perder tarjeta
- Por quedarse todo el día

Confirmar Pago

Cobrar tarjeta

Cobrar	
Identificador de tarjeta: <input type="text" value="xxxxx"/> <input type="button" value="Mostrar"/>	
Fecha de Salida: XX-XX-XXXX Hora Actual: XX Hora de Salida: XX:XX:XX Total de minutos: XXX Precio Total: XXX MXN	
<input type="button" value="Asignar Multa(s) ▾"/> <input type="button" value="Asignar Método de Pago ▾"/> <div style="display: flex; justify-content: space-between;"> <div style="flex: 1; text-align: center;"> <input type="button" value="Tarjeta de crédito/débito"/> </div> <div style="flex: 1; text-align: center;"> <input type="button" value="Efectivo"/> </div> </div> <input type="button" value="Confirmar Pago"/>	

Cobrar tarjeta

Cobrar	
Identificador de tarjeta: <input type="text" value="xxxxx"/> <input type="button" value="Mostrar"/>	
Fecha de Salida: XX-XX-XXXX Hora Actual: XX Hora de Salida: XX:XX:XX Total de minutos: XXX ¡El pago se ha acreditado de manera exitosa!	
<input type="button" value="Asignar Multa(s) ▾"/> <div style="display: flex; justify-content: space-between;"> <div style="flex: 1; text-align: center;"> <input type="button" value="Asignar Multa(s) ▾"/> </div> <div style="flex: 1; text-align: center;"> <input type="button" value="Asignar Método de Pago ▾"/> </div> </div> <div style="margin-top: 10px;"> <input type="button" value="OK"/> </div> <input type="button" value="Confirmar Pago"/>	

CU03– Administrar costos

Precio por horas para automóviles	Precio por horas para motos	Precio de multas
Modificar		
Horas	Tarifa	Descripción
X	XXX	XXXXXXXXXXXXXXXXXXXX

Precio por horas para automóviles	Precio por horas para motos	Precio de multas	
Modificar			
Horas	Tarifa	Descripción	
X	Modificar		
X	Horas	Tarifa	Descripción
X	XX	XXX	XXXXXXXXXXXXXXXXXXXX
X			
X	Cancelar Actualizar		
X	XXX	XXXXXXXXXXXXXXXXXXXX	
X	XXX	XXXXXXXXXXXXXXXXXXXX	
X	XXX	XXXXXXXXXXXXXXXXXXXX	

Precio por horas para automóviles	Precio por horas para motos	Precio de multas
Modificar		
Horas	Tarifa	Descripción
X	Modificar	
X	Modificación Realizada	
X	La tarifa ha sido modificada exitosamente	
X		
X	Confirmar	
X	XXX	XXXXXXXXXXXXXXXXXXXX
X	XXX	XXXXXXXXXXXXXXXXXXXX
X	XXX	XXXXXXXXXXXXXXXXXXXX

Precio por horas para automóviles	Precio por horas para motos	Precio de multas
Modificar		
Horas	Tarifa	Descripción
X	XXX	XXXXXXXXXXXXXXXXXXXX

Precio por horas para automóviles	Precio por horas para motos	Precio de multas																														
<input type="button" value="Modificar"/>																																
<table border="1"> <thead> <tr> <th>Horas</th> <th>Tarifa</th> <th>Descripción</th> </tr> </thead> <tbody> <tr> <td>X</td> <td colspan="2">Modificar</td> </tr> <tr> <td>X</td> <td>Horas</td> <td>Tarifa</td> </tr> <tr> <td>X</td> <td>XX</td> <td>XXX</td> </tr> <tr> <td>X</td> <td colspan="2">Descripción</td> </tr> <tr> <td>X</td> <td colspan="2">XXXXXXXXXXXXXXXXXXXX</td> </tr> <tr> <td>X</td> <td colspan="2"> <input type="button" value="Cancelar"/> <input type="button" value="Actualizar"/> </td> </tr> <tr> <td>X</td> <td>XXX</td> <td>XXXXXXXXXXXXXXXXXXXX</td> </tr> <tr> <td>X</td> <td>XXX</td> <td>XXXXXXXXXXXXXXXXXXXX</td> </tr> <tr> <td>X</td> <td>XXX</td> <td>XXXXXXXXXXXXXXXXXXXX</td> </tr> </tbody> </table>			Horas	Tarifa	Descripción	X	Modificar		X	Horas	Tarifa	X	XX	XXX	X	Descripción		X	XXXXXXXXXXXXXXXXXXXX		X	<input type="button" value="Cancelar"/> <input type="button" value="Actualizar"/>		X	XXX	XXXXXXXXXXXXXXXXXXXX	X	XXX	XXXXXXXXXXXXXXXXXXXX	X	XXX	XXXXXXXXXXXXXXXXXXXX
Horas	Tarifa	Descripción																														
X	Modificar																															
X	Horas	Tarifa																														
X	XX	XXX																														
X	Descripción																															
X	XXXXXXXXXXXXXXXXXXXX																															
X	<input type="button" value="Cancelar"/> <input type="button" value="Actualizar"/>																															
X	XXX	XXXXXXXXXXXXXXXXXXXX																														
X	XXX	XXXXXXXXXXXXXXXXXXXX																														
X	XXX	XXXXXXXXXXXXXXXXXXXX																														
<input type="button" value="Modificar"/> <input type="button" value="Añadir"/> <input type="button" value="Eliminar"/>																																
<table border="1"> <thead> <tr> <th>Horas</th> <th>Tarifa</th> <th>Descripción</th> </tr> </thead> <tbody> <tr> <td>X</td> <td colspan="2">Modificar</td> </tr> <tr> <td>X</td> <td colspan="2">Modificación Realizada</td> </tr> <tr> <td>X</td> <td colspan="2">La tarifa ha sido modificada exitosamente</td> </tr> <tr> <td>X</td> <td colspan="2"> <input type="button" value="Confirmar"/> </td> </tr> <tr> <td>X</td> <td>XXX</td> <td>XXXXXXXXXXXXXXXXXXXX</td> </tr> <tr> <td>X</td> <td>XXX</td> <td>XXXXXXXXXXXXXXXXXXXX</td> </tr> <tr> <td>X</td> <td>XXX</td> <td>XXXXXXXXXXXXXXXXXXXX</td> </tr> </tbody> </table>			Horas	Tarifa	Descripción	X	Modificar		X	Modificación Realizada		X	La tarifa ha sido modificada exitosamente		X	<input type="button" value="Confirmar"/>		X	XXX	XXXXXXXXXXXXXXXXXXXX	X	XXX	XXXXXXXXXXXXXXXXXXXX	X	XXX	XXXXXXXXXXXXXXXXXXXX						
Horas	Tarifa	Descripción																														
X	Modificar																															
X	Modificación Realizada																															
X	La tarifa ha sido modificada exitosamente																															
X	<input type="button" value="Confirmar"/>																															
X	XXX	XXXXXXXXXXXXXXXXXXXX																														
X	XXX	XXXXXXXXXXXXXXXXXXXX																														
X	XXX	XXXXXXXXXXXXXXXXXXXX																														
Precio por horas para automóviles	Precio por horas para motos	Precio de multas																														
<input type="button" value="Modificar"/> <input type="button" value="Añadir"/> <input type="button" value="Eliminar"/>																																
<table border="1"> <thead> <tr> <th>Titulo</th> <th>Tarifa</th> <th>Descripción</th> </tr> </thead> <tbody> <tr> <td>XXXXX</td> <td>XXX</td> <td>XXXXXXXXXXXXXXXXXXXX</td> </tr> </tbody> </table>			Titulo	Tarifa	Descripción	XXXXX	XXX	XXXXXXXXXXXXXXXXXXXX	XXXXX	XXX	XXXXXXXXXXXXXXXXXXXX	XXXXX	XXX	XXXXXXXXXXXXXXXXXXXX	XXXXX	XXX	XXXXXXXXXXXXXXXXXXXX	XXXXX	XXX	XXXXXXXXXXXXXXXXXXXX	XXXXX	XXX	XXXXXXXXXXXXXXXXXXXX	XXXXX	XXX	XXXXXXXXXXXXXXXXXXXX	XXXXX	XXX	XXXXXXXXXXXXXXXXXXXX			
Titulo	Tarifa	Descripción																														
XXXXX	XXX	XXXXXXXXXXXXXXXXXXXX																														
XXXXX	XXX	XXXXXXXXXXXXXXXXXXXX																														
XXXXX	XXX	XXXXXXXXXXXXXXXXXXXX																														
XXXXX	XXX	XXXXXXXXXXXXXXXXXXXX																														
XXXXX	XXX	XXXXXXXXXXXXXXXXXXXX																														
XXXXX	XXX	XXXXXXXXXXXXXXXXXXXX																														
XXXXX	XXX	XXXXXXXXXXXXXXXXXXXX																														
XXXXX	XXX	XXXXXXXXXXXXXXXXXXXX																														

Precio por horas para automóviles	Precio por horas para motos	Precio de multas																														
Modificar	Añadir	Eliminar																														
<table border="1"> <thead> <tr> <th>Titulo</th> <th>Tarifa</th> <th>Descripción</th> </tr> </thead> <tbody> <tr> <td>XXXXX</td> <td colspan="2">Añadir Tarifa</td> </tr> <tr> <td>XXXXX</td> <td>Titulo XXXXX</td> <td>Tarifa XXX</td> <td>Descripción XXXXXXXXXXXXXXXXXX</td> </tr> <tr> <td>XXXXX</td> <td colspan="3"></td> </tr> <tr> <td>XXXXX</td> <td colspan="3"> <input type="button" value="Cancelar"/> <input type="button" value="Confirmar"/> </td> </tr> <tr> <td>XXXXX</td> <td>XXX</td> <td colspan="2">XXXXXXXXXXXXXXXXXXXXXX</td> </tr> <tr> <td>XXXXX</td> <td>XXX</td> <td colspan="2">XXXXXXXXXXXXXXXXXXXXXX</td> </tr> <tr> <td>XXXXX</td> <td>XXX</td> <td colspan="2">XXXXXXXXXXXXXXXXXXXXXX</td> </tr> </tbody> </table>			Titulo	Tarifa	Descripción	XXXXX	Añadir Tarifa		XXXXX	Titulo XXXXX	Tarifa XXX	Descripción XXXXXXXXXXXXXXXXXX	XXXXX				XXXXX	<input type="button" value="Cancelar"/> <input type="button" value="Confirmar"/>			XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX		XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX		XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX	
Titulo	Tarifa	Descripción																														
XXXXX	Añadir Tarifa																															
XXXXX	Titulo XXXXX	Tarifa XXX	Descripción XXXXXXXXXXXXXXXXXX																													
XXXXX																																
XXXXX	<input type="button" value="Cancelar"/> <input type="button" value="Confirmar"/>																															
XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX																														
XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX																														
XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX																														

Precio por horas para automóviles	Precio por horas para motos	Precio de multas																								
Modificar	Añadir	Eliminar																								
<table border="1"> <thead> <tr> <th>Titulo</th> <th>Tarifa</th> <th>Descripción</th> </tr> </thead> <tbody> <tr> <td>XXXXX</td> <td colspan="2">Añadir Tarifa</td> </tr> <tr> <td>XXXXX</td> <td colspan="2">Tarifa Registrada</td> </tr> <tr> <td>XXXXX</td> <td colspan="2">Se ha añadido una nueva tarifa de pago</td> </tr> <tr> <td>XXXXX</td> <td colspan="2"> <input type="button" value="Confirmar"/> </td> </tr> <tr> <td>XXXXX</td> <td>XXX</td> <td>XXXXXXXXXXXXXXXXXXXXXX</td> </tr> <tr> <td>XXXXX</td> <td>XXX</td> <td>XXXXXXXXXXXXXXXXXXXXXX</td> </tr> <tr> <td>XXXXX</td> <td>XXX</td> <td>XXXXXXXXXXXXXXXXXXXXXX</td> </tr> </tbody> </table>			Titulo	Tarifa	Descripción	XXXXX	Añadir Tarifa		XXXXX	Tarifa Registrada		XXXXX	Se ha añadido una nueva tarifa de pago		XXXXX	<input type="button" value="Confirmar"/>		XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX	XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX	XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX
Titulo	Tarifa	Descripción																								
XXXXX	Añadir Tarifa																									
XXXXX	Tarifa Registrada																									
XXXXX	Se ha añadido una nueva tarifa de pago																									
XXXXX	<input type="button" value="Confirmar"/>																									
XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX																								
XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX																								
XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX																								

Precio por horas para automóviles	Precio por horas para motos	Precio de multas																														
Modificar	Añadir	Eliminar																														
<table border="1"> <thead> <tr> <th>Titulo</th> <th>Tarifa</th> <th>Descripción</th> </tr> </thead> <tbody> <tr> <td>XXXXX</td> <td colspan="2">Modificar</td> </tr> <tr> <td>XXXXX</td> <td>Titulo XXXXX</td> <td>Tarifa XXX</td> <td>Descripción XXXXXXXXXXXXXXXXXX</td> </tr> <tr> <td>XXXXX</td> <td colspan="3"></td> </tr> <tr> <td>XXXXX</td> <td colspan="3"> <input type="button" value="Cancelar"/> <input type="button" value="Actualizar"/> </td> </tr> <tr> <td>XXXXX</td> <td>XXX</td> <td colspan="2">XXXXXXXXXXXXXXXXXXXXXX</td> </tr> <tr> <td>XXXXX</td> <td>XXX</td> <td colspan="2">XXXXXXXXXXXXXXXXXXXXXX</td> </tr> <tr> <td>XXXXX</td> <td>XXX</td> <td colspan="2">XXXXXXXXXXXXXXXXXXXXXX</td> </tr> </tbody> </table>			Titulo	Tarifa	Descripción	XXXXX	Modificar		XXXXX	Titulo XXXXX	Tarifa XXX	Descripción XXXXXXXXXXXXXXXXXX	XXXXX				XXXXX	<input type="button" value="Cancelar"/> <input type="button" value="Actualizar"/>			XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX		XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX		XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX	
Titulo	Tarifa	Descripción																														
XXXXX	Modificar																															
XXXXX	Titulo XXXXX	Tarifa XXX	Descripción XXXXXXXXXXXXXXXXXX																													
XXXXX																																
XXXXX	<input type="button" value="Cancelar"/> <input type="button" value="Actualizar"/>																															
XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX																														
XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX																														
XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX																														

Precio por horas para automóviles	Precio por horas para motos	Precio de multas																								
Modificar	Añadir	Eliminar																								
<table border="1"> <thead> <tr> <th>Titulo</th> <th>Tarifa</th> <th>Descripción</th> </tr> </thead> <tbody> <tr> <td>XXXXX</td> <td colspan="2">Modificar</td> </tr> <tr> <td>XXXXX</td> <td colspan="2">Modificación Realizada</td> </tr> <tr> <td>XXXXX</td> <td colspan="2">La tarifa ha sido modificada exitosamente</td> </tr> <tr> <td>XXXXX</td> <td colspan="2">Confirmar</td> </tr> <tr> <td>XXXXX</td> <td>XXX</td> <td>XXXXXXXXXXXXXXXXXXXXXX</td> </tr> <tr> <td>XXXXX</td> <td>XXX</td> <td>XXXXXXXXXXXXXXXXXXXXXX</td> </tr> <tr> <td>XXXXX</td> <td>XXX</td> <td>XXXXXXXXXXXXXXXXXXXXXX</td> </tr> </tbody> </table>			Titulo	Tarifa	Descripción	XXXXX	Modificar		XXXXX	Modificación Realizada		XXXXX	La tarifa ha sido modificada exitosamente		XXXXX	Confirmar		XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX	XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX	XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX
Titulo	Tarifa	Descripción																								
XXXXX	Modificar																									
XXXXX	Modificación Realizada																									
XXXXX	La tarifa ha sido modificada exitosamente																									
XXXXX	Confirmar																									
XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX																								
XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX																								
XXXXX	XXX	XXXXXXXXXXXXXXXXXXXXXX																								

CU04– Administrar tarjetas

< Administrar registros y tarjetas

Registros	Tarjetas																											
Registros del día: 15-05-2023 <div style="float: right;">Cambiar fecha</div> <table border="1"> <thead> <tr> <th>Registro #001</th> <th>Fecha y hora de entrada: XX:XX XX/XX/XXXX</th> <th>Tarifa: XXXX</th> </tr> </thead> <tbody> <tr> <td>ID Tarjeta: XXXX No. Cajón: XXXX Tipo de vehículo: XXXX</td> <td>Fecha y hora de salida: XX:XX XX/XX/XXXX Tiempo transcurrido: XX minutos</td> <td>Estatus de la tarifa: XXXX Método de pago: XXXX</td> </tr> <tr> <td>Modificar</td> <td></td> <td></td> </tr> </tbody> </table> <table border="1"> <thead> <tr> <th>Registro #002</th> <th>Fecha y hora de entrada: XX:XX XX/XX/XXXX</th> <th>Tarifa: XXXX</th> </tr> </thead> <tbody> <tr> <td>ID Tarjeta: XXXX No. Cajón: XXXX Tipo de vehículo: XXXX</td> <td>Fecha y hora de salida: XX:XX XX/XX/XXXX Tiempo transcurrido: XX minutos</td> <td>Estatus de la tarifa: XXXX Método de pago: XXXX</td> </tr> <tr> <td>Modificar</td> <td></td> <td></td> </tr> </tbody> </table> <table border="1"> <thead> <tr> <th>Registro #003</th> <th>Fecha y hora de entrada: XX:XX XX/XX/XXXX</th> <th>Tarifa: XXXX</th> </tr> </thead> <tbody> <tr> <td>ID Tarjeta: XXXX No. Cajón: XXXX Tipo de vehículo: XXXX</td> <td>Fecha y hora de salida: XX:XX XX/XX/XXXX Tiempo transcurrido: XX minutos</td> <td>Estatus de la tarifa: XXXX Método de pago: XXXX</td> </tr> <tr> <td>Modificar</td> <td></td> <td></td> </tr> </tbody> </table>		Registro #001	Fecha y hora de entrada: XX:XX XX/XX/XXXX	Tarifa: XXXX	ID Tarjeta: XXXX No. Cajón: XXXX Tipo de vehículo: XXXX	Fecha y hora de salida: XX:XX XX/XX/XXXX Tiempo transcurrido: XX minutos	Estatus de la tarifa: XXXX Método de pago: XXXX	Modificar			Registro #002	Fecha y hora de entrada: XX:XX XX/XX/XXXX	Tarifa: XXXX	ID Tarjeta: XXXX No. Cajón: XXXX Tipo de vehículo: XXXX	Fecha y hora de salida: XX:XX XX/XX/XXXX Tiempo transcurrido: XX minutos	Estatus de la tarifa: XXXX Método de pago: XXXX	Modificar			Registro #003	Fecha y hora de entrada: XX:XX XX/XX/XXXX	Tarifa: XXXX	ID Tarjeta: XXXX No. Cajón: XXXX Tipo de vehículo: XXXX	Fecha y hora de salida: XX:XX XX/XX/XXXX Tiempo transcurrido: XX minutos	Estatus de la tarifa: XXXX Método de pago: XXXX	Modificar		
Registro #001	Fecha y hora de entrada: XX:XX XX/XX/XXXX	Tarifa: XXXX																										
ID Tarjeta: XXXX No. Cajón: XXXX Tipo de vehículo: XXXX	Fecha y hora de salida: XX:XX XX/XX/XXXX Tiempo transcurrido: XX minutos	Estatus de la tarifa: XXXX Método de pago: XXXX																										
Modificar																												
Registro #002	Fecha y hora de entrada: XX:XX XX/XX/XXXX	Tarifa: XXXX																										
ID Tarjeta: XXXX No. Cajón: XXXX Tipo de vehículo: XXXX	Fecha y hora de salida: XX:XX XX/XX/XXXX Tiempo transcurrido: XX minutos	Estatus de la tarifa: XXXX Método de pago: XXXX																										
Modificar																												
Registro #003	Fecha y hora de entrada: XX:XX XX/XX/XXXX	Tarifa: XXXX																										
ID Tarjeta: XXXX No. Cajón: XXXX Tipo de vehículo: XXXX	Fecha y hora de salida: XX:XX XX/XX/XXXX Tiempo transcurrido: XX minutos	Estatus de la tarifa: XXXX Método de pago: XXXX																										
Modificar																												

< Administrar registros y tarjetas

Registros	Tarjetas																																																	
Registros del día: 15- <div style="float: right;"> Cambiar fecha <div style="border: 1px solid black; padding: 5px; margin-top: 5px;"> MAYO - 2023 <table border="1" style="margin-top: 5px; font-size: small;"> <tr> <th>S</th><th>M</th><th>T</th><th>W</th><th>T</th><th>F</th><th>Sa</th></tr> <tr> <td>30</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td></tr> <tr> <td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td></tr> <tr> <td>14</td><td>15</td><td>16</td><td>17</td><td>18</td><td>19</td><td>20</td></tr> <tr> <td>21</td><td>22</td><td>23</td><td>24</td><td>25</td><td>26</td><td>27</td></tr> <tr> <td>28</td><td>29</td><td>30</td><td>31</td><td>1</td><td>2</td><td>3</td></tr> <tr> <td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td></tr> </table> <div style="margin-top: 10px;"> Seleccionar y ver registros </div> </div> </div>		S	M	T	W	T	F	Sa	30	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	1	2	3	4	5	6	7	8	9	10
S	M	T	W	T	F	Sa																																												
30	1	2	3	4	5	6																																												
7	8	9	10	11	12	13																																												
14	15	16	17	18	19	20																																												
21	22	23	24	25	26	27																																												
28	29	30	31	1	2	3																																												
4	5	6	7	8	9	10																																												
<table border="1"> <thead> <tr> <th>Registro #001</th> </tr> </thead> <tbody> <tr> <td>ID Tarjeta: XXXX No. Cajón: XXXX Tipo de vehículo: XXXX</td> </tr> <tr> <td>Modificar</td> </tr> </tbody> </table> <table border="1"> <thead> <tr> <th>Registro #002</th> </tr> </thead> <tbody> <tr> <td>ID Tarjeta: XXXX No. Cajón: XXXX Tipo de vehículo: XXXX</td> </tr> <tr> <td>Modificar</td> </tr> </tbody> </table> <table border="1"> <thead> <tr> <th>Registro #003</th> </tr> </thead> <tbody> <tr> <td>ID Tarjeta: XXXX No. Cajón: XXXX Tipo de vehículo: XXXX</td> </tr> <tr> <td>Modificar</td> </tr> </tbody> </table>	Registro #001	ID Tarjeta: XXXX No. Cajón: XXXX Tipo de vehículo: XXXX	Modificar	Registro #002	ID Tarjeta: XXXX No. Cajón: XXXX Tipo de vehículo: XXXX	Modificar	Registro #003	ID Tarjeta: XXXX No. Cajón: XXXX Tipo de vehículo: XXXX	Modificar																																									
Registro #001																																																		
ID Tarjeta: XXXX No. Cajón: XXXX Tipo de vehículo: XXXX																																																		
Modificar																																																		
Registro #002																																																		
ID Tarjeta: XXXX No. Cajón: XXXX Tipo de vehículo: XXXX																																																		
Modificar																																																		
Registro #003																																																		
ID Tarjeta: XXXX No. Cajón: XXXX Tipo de vehículo: XXXX																																																		
Modificar																																																		

◀ Administrar registros y tarjetas

Registros

Tarjetas

Registros del día: 0

Modificar registro #XXX

Registro #128

ID Tarjeta: XXX

No. Cajón: XXX

Tipo de vehículo: XXXX

Registro #129

ID Tarjeta: XXX

No. Cajón: XXX

Tipo de vehículo: XXXX

Registro #130

ID Tarjeta: XXX

No. Cajón: XXX

Tipo de vehículo: XXXX

ID Tarjeta:

XXX

Hora de entrada:

XX:XX:XX

Fecha de entrada:

XX-XX-XXXX

Tarifa:

XXX

No. Cajón:

XXX

Hora de salida:

XX:XX:XX

Fecha de salida:

XX-XX-XXXX

Estatus de tarifa:

XXX

Tipo de vehículo:

XXXX

Tiempo transcurrido:

XX minutos

Método de pago:

XXXX

Estatus de registro:

XXXX

Cancelar

Guardar cambios

Modificar

Modificar

Modificar

◀ Administrar registros y tarjetas

Registros

Tarjetas



Buscar

Guardar cambios

◀ Administrar registros y tarjetas

Registros

Tarjetas

XXXXX



Buscar

Código: XXXXX

Cajón: XX

Nivel: XX

Estatus de la tarjeta:

XXXX

Guardar cambios

4. Características del sistema

Descripción

El Sistema de Estacionamiento JETS es un sistema de gestión de estacionamiento, el cual sirve para automatizar algunas de las actividades realizadas y facilitar otras. En el sistema, se podrá checar la disponibilidad del estacionamiento, guardar registros con base a las tarjetas proporcionadas, generar el costo total a pagar del cliente, asignar multas, producir reportes con base a los registros, y modificar las tarifas del estacionamiento.

El sistema va dirigido principalmente a los empleados del estacionamiento, ya que estos interactuarán directamente con el sistema; sin embargo, también se verá favorecido los clientes, ya que se facilitará el proceso de proporcionar tarjetas y realizar los cobros.

El sistema será una aplicación implementada en el lenguaje de programación de Java ocupando el entorno de desarrollo NetBeans.

Requerimientos funcionales

En el siguiente apartado se describen los requerimientos funcionales que el sistema deberá cumplir:

- PropTarj.RQF01. El sistema debe permitir al despachador checar la disponibilidad del estacionamiento por niveles.
- PropTarj.RQF02. El sistema debe permitir el actualizar automáticamente la disponibilidad del estacionamiento, si el sistema de sensores comunica un cambio en el estado de un cajón en el estacionamiento
- PropTarj.RQF03. El sistema debe permitir el mostrar un mensaje indicando que no se admiten más vehículos al estacionamiento si no hay disponibilidad en el estacionamiento
- PropTarj.RQF04. El sistema debe permitir el asignar automáticamente el tipo de vehículo, la hora y fecha de entrada del cliente en el registro cuando se indica que se entrega una tarjeta a este.
- PropTarj.RQF05. El sistema debe permitir el iniciar a tomar el tiempo que está el cliente en el estacionamiento en el registro desde que se proporciona la tarjeta a este.
- CobrTarj.RQF01. El sistema debe permitir el indicar la tarjeta a la que se le realizará el cobro.
- CobrTarj.RQF02. El sistema debe permitir el asignar una multa al cliente
- CobrTarj.RQF03. El sistema debe permitir el generar automáticamente el costo total a pagar con base a la información proporcionada de los registros.
- CobrTarj.RQF04. El sistema debe permitir registrar los métodos de pago como efectivo o con tarjeta.

- CobrTarj.RQF05. El sistema debe permitir el indicar que se devolvió la tarjeta del cajón cuando se registra como pagado.
- CobrTarj.RQF06. El sistema debe permitir el visualizar los detalles del pago.
- CobrTarj.RQF07. El sistema debe permitir el actualizar los registros al realizar un cobro.
- CobrTarj.RQF08. El sistema debe permitir indicar el estado de la tarjeta como “Perdido” cuando se pierde una tarjeta.
- AdmCosto.RQF01. El sistema debe permitir el estipular tarifas de pago en el sistema de gestión de estacionamiento en función del tipo de vehículo, es decir, el tipo de cajón utilizado
- AdmCosto.RQF02. El sistema debe permitir el estipular tarifas de pago en el sistema de gestión de estacionamiento en función de la duración de la estadía
- AdmCosto.RQF03. El sistema debe permitir el estipular costos de multas en el sistema de gestión de estacionamiento determinadas por el uso inadecuado de un cajón de estacionamiento
- AdmCosto.RQF04. El sistema debe permitir el estipular costos de multas en el sistema de gestión de estacionamiento determinadas por el estacionamiento de un vehículo que excede las 13 horas permitidas del estacionamiento
- AdmCosto.RQF05. El sistema debe permitir el estipular costos de multas en el sistema de gestión de estacionamiento determinadas por perder la tarjeta del cajón asignado al ingresar al estacionamiento
- AdmUsu.RQF01. El sistema debe permitir el registrar a un usuario como tipo despachador.
- AdmUsu.RQF02. El sistema debe permitir el registrar a un usuario como tipo administrador.
- AdmUsu.RQF03. El sistema debe permitir el modificar los datos de un usuario.
- AdmUsu.RQF04. El sistema debe permitir el eliminar un usuario registrado previamente.
- AdmUsu.RQF05. El sistema debe permitir el consultar los usuarios registrados.
- IniSes.RQF01. El sistema debe permitir el acceso al usuario despachador al sistema durante su jornada laboral.
- IniSes.RQF02. El sistema debe permitir el ingresar al usuario administrador al sistema a cualquier momento del día exceptuando horas de mantenimiento.
- AdmTarjReg.RQF01. El sistema debe permitir al administrador visualizar los datos de los registros y de las tarjetas.
- AdmTarjReg.RQF02. El sistema debe permitir al administrador modificar los estatus de los registros.
- AdmTarjReg.RQF03. El sistema debe permitir al administrador modificar los estatus de las tarjetas.

- GenRep.RQF01. El sistema debe permitir el generar un reporte que incluya los detalles de todos los registros de las tarjetas expedidas durante el día.
- GenRepor.RQF02. El sistema debe permitir generar reportes de las tarjetas en formato PDF.
- GenRep.RQF03. El sistema debe permitir el descargar los reportes generados.
- GenRepor.RQF04. El sistema debe permitir al administrador cambiar el día de generación de reportes.

5. Requerimientos no funcionales

Atributos de calidad del software

Los atributos de calidad que se considerarán serán:

Disponibilidad:

El software debe estar disponible para su uso únicamente durante las horas hábiles de la compañía; esto se determinará por el tipo de usuario que desee ingresar al sistema. Para el cumplimiento de dicho requerimiento se necesitará:

Horarios de acceso que permita únicamente el registro de usuarios autorizados dentro de sus horarios hábiles, en este caso el administrador podrá acceder al sistema en cualquier jornada laboral, excepto de 12AM a 2AM (Horas de mantenimiento) y el despachador podrá acceder al sistema solamente en las horas validas, es decir, en su jornada laboral de 7AM a 8PM

En caso de que se intente ingresar en horas inhábiles el sistema mostrara una alerta a los usuarios que mostrara los horarios hábiles del software y evitara el ingreso al sistema

Cuando se necesite un mantenimiento del sistema este se deberá programar fuera de las horas hábiles para evitar el chocar con horas donde el sistema deberá estar operativo y funcionando

Desempeño:

El software debe cumplir con todas las solicitudes que hagan todos los usuarios que hacen uso de este sistema, sin indiferencia alguna. Además, el sistema deberá tener un tiempo de respuesta de máximo de 10 segundos durante los horarios hábiles de la compañía y los de mayor concurrencia de clientes. Esto se debe a que, normalmente, 2 usuarios estarán haciendo uso del sistema en tiempo real, por lo que se espera que se pueda mantener el tiempo de respuesta predilecto.

No obstante, se prevé que en los horarios de mayor demanda de clientes y, por ende, de mayor tráfico de usuarios haciendo uso del sistema, se mantenga el tiempo de respuesta asignado o inclusive que incremente a un tiempo de 15 segundos como máximo.

Mantenibilidad:

El sistema será configurado e irá implementando correcciones para arreglar errores inesperados, o simplemente para ir actualizando el sistema cuando sea necesario.

Durante las actualizaciones, el sistema no estará disponible para evitar algún fallo al momento de ejecutar acciones. Por lo que estas correcciones se llevarán a cabo fuera del horario de servicio al público. A su vez, en caso de que se detecte un error en el sistema, se tendrá que programar una reunión para corregirlo.

Seguridad e Integridad:

El sistema protegerá contra el acceso no autorizado a este y sus datos al implementar el uso de cuentas por cada individuo que deseé ingresar y tenga permiso para entrar. Esta cuenta estará conformada por un usuario y contraseña, la cual debe tener una longitud mínima de 8 caracteres para ser aceptada por el sistema.

Además, para asegurar que cada usuario tenga acceso a las funcionalidades y datos que le corresponden, al registrar una cuenta se deberá indicar qué tipo de usuario se está registrando. Habrá 2 roles principales: Despachador y Administrador.

En cuanto a la integridad, se implementará en la base de datos la restricción de dominios y de valores únicos.

La restricción de dominios se logrará al restringir los valores no nulos en las claves primarias de cada tabla, para que así, el registro siempre tenga una clave para identificarla, aunque falten otros atributos del registro. Y la restricción de valores únicos En este caso se añadió esta restricción para garantizar que no se inserten valores duplicados en una columna específica, y así asegurar la integridad de los datos.

Reglas de negocio

Hechos

RNH01 - El horario laboral del estacionamiento es de 7am a 8pm para despachadores.

RNH02 - Cada tarjeta corresponde a un cajón en específico.

RNH03 - El administrador repone las tarjetas perdidas.

RNH04 – El tiempo que se le cobra al cliente se toma desde que se le proporciona su tarjeta hasta que inicia el proceso de pago.

RNH05 – Los costos de las tarifas se pueden modificar por el administrador.

RNH06 – Hay 4 niveles para automóviles y 1 nivel para motos, cada uno con 50 cajones.

Restricciones

RNR01 – El despachador es el único que puede proporcionar tarjetas y realizar el cobro de estas.

RNR02 – El administrador es el único que puede administrar los usuarios, las tarjetas y los costos.

RNR03 – El despachador sólo puede acceder al sistema dentro de su jornada laboral.

RNR04 – Una vez que se crea su cuenta, no se puede cambiar su tipo de usuario.

RNR05 – El administrador no puede acceder al sistema en horario de mantenimiento

Habilitadores de acción

RNHA01 – Si el cliente pierde la tarjeta, entonces se le cobrará una multa.

RNHA03 - Si el cliente se queda después de la hora de cierre, entonces se le cobrará una multa.

RNHA04 - Si el estacionamiento de queda sin lugares, entonces se muestra el mensaje “Cupo lleno”.

RNHA05 – Si el despachador indica que se entrega una tarjeta, entonces no se puede entregar la misma tarjeta a otro cliente.

6. Planeación

Cronograma de actividades

Cronograma de actividades			
Elemento	Contenido	Duración	Fechas
Introducción	Propósito	4 días	2 al 5 de mayo
	Alcance del producto		
Descripción general	Modelo de proceso empleado	4 días	2 al 5 de mayo
	Funciones del producto		
	Clases de usuario y características		
	Diagramas de Eriksson Penker y de BPMN		
Características del sistema	Descripción	1 día	6 de mayo
	Requerimientos funcionales		
Otros requerimientos no funcionales	Atributos de calidad del software	3 días	7 al 9 de mayo
	Reglas del negocio		
Análisis de requerimientos	Diagrama de Casos de Uso	3 días	10 al 12 de mayo
	Descripción de los Casos de Uso		
Requerimientos de interfaz externa	Interfaces de usuario	1 día	13 de mayo
Diseño	Diagrama Entidad-Relación	4 días	14 al 17 de mayo
	Diagrama Relacional		
	Diagrama de Clases	3 días	18 al 20 de mayo
Implementación	Implementación de la Base de datos	2 días	21 al 22 de mayo
	Codificación	1 semana y 4 días	23 de mayo al 3 de junio
Pruebas de la implementación	Pruebas unitarias	3 días	31 de mayo al 2 de junio
	Pruebas generales		
Conclusión	Conclusión	1 día	4 de junio

Tabla 6.1.- Cronograma de actividades

7. Análisis de requerimientos

En este apartado se muestra el diagrama de caso de uso del sistema y las descripciones para los casos de uso que se implementarán.

Diagramas de casos de uso

En el siguiente diagrama se muestran los casos de uso de todo el sistema

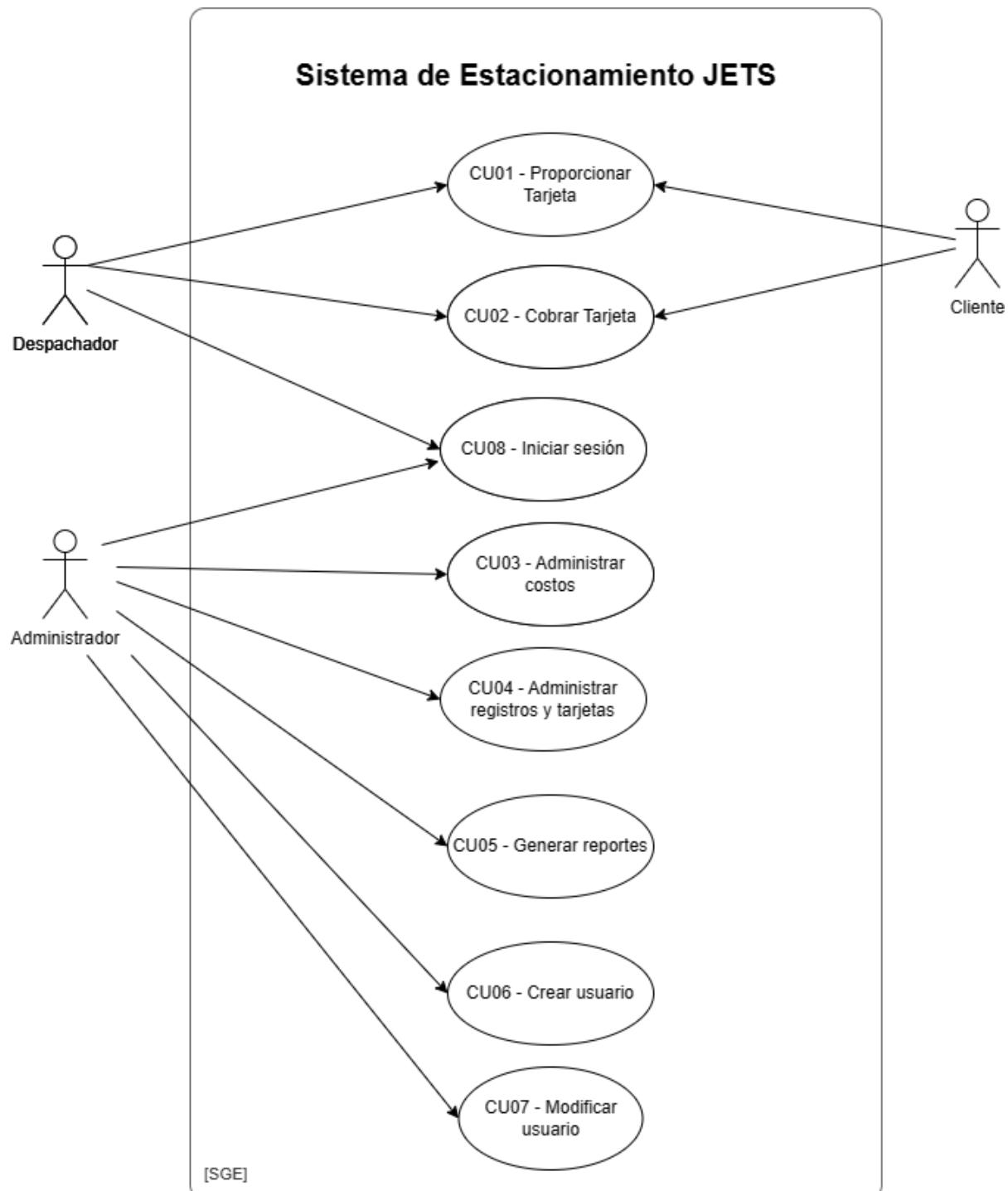


Ilustración 7 Diagrama de casos de uso

Descripción de los casos de uso

Caso de uso 1: Proporcionar tarjeta

ID:	CU01
Nombre del CU:	<i>Proporcionar tarjeta</i>
Responsable:	Sulem Martínez Aguilar
Fecha de actualización:	10/05/2023
Descripción:	El despachador muestra la disponibilidad al cliente para que este seleccione un nivel del estacionamiento y con base a esto le pueda proporcionar una tarjeta correspondiente a un cajón del nivel seleccionado
Actor(es):	- Despachador - Cliente
Disparador:	El despachador selecciona “Despachar tarjeta” en el menú principal
Precondiciones:	PRE01. El despachador debe tener una sesión iniciada con su cuenta en el sistema.
Flujo Normal:	<ol style="list-style-type: none"> 1. El sistema muestra la disponibilidad actual del estacionamiento iniciando por el primer nivel y un botón para indicar que se proporcionará una tarjeta del nivel especificado (Ver FA1.1, EX01). 2. El despachador le muestra la disponibilidad al cliente y este indica qué nivel quiere. 3. El despachador selecciona otro nivel para ver la disponibilidad (Ver FA3.1) 4. El sistema muestra la disponibilidad del nivel seleccionado. 5. El despachador selecciona “Dar tarjeta del nivel”. 6. El sistema muestra la tarjeta, correspondiente a un cajón del nivel seleccionado, que se le debe dar al cliente y un mensaje para confirmar que se entregará la tarjeta indicada. 7. El despachador selecciona “Confirmar” (Ver FA7.1). 8. El sistema genera un registro con la tarjeta, el tipo de vehículo, la hora y la fecha actuales en la base de datos. 9. El sistema muestra los cajones actualizados y muestra “Tarjeta proporcionada”. 10. Termina caso de uso.
Flujos Alternos:	<p>FA1.1. No hay espacios disponibles en todo el estacionamiento</p> <ol style="list-style-type: none"> 1. El sistema muestra el mensaje “Cupo lleno” 2. Termina caso de uso. <p>FA3.1. El cliente elige quedarse en el nivel inicial</p> <ol style="list-style-type: none"> 1. Ir al paso 5 (Flujo normal). <p>FA7.1. El despachador cancela la entrega de la tarjeta.</p> <ol style="list-style-type: none"> 1. El despachador selecciona “Cancelar”. 2. Ir al paso 4 (Flujo normal).
Excepciones:	<p>EX01. Hubo un error con la conexión a la base de datos</p> <ol style="list-style-type: none"> 1. El sistema muestra el mensaje “No se pudo conectar con la base de datos. Inténtelo de nuevo o hágalo más tarde”. 2. Termina caso de uso.
Postcondiciones:	<p>POST01 - El sistema genera un registro con la tarjeta, el tipo de vehículo, la hora y la fecha actuales en la base de datos.</p> <p>POST02 - El sistema muestra los cajones actualizados y el mensaje de “Tarjeta proporcionada”.</p>

Reglas de negocio:	RNH02 - Cada tarjeta corresponde a un cajón en específico. RNR01 – El despachador es el único que puede proporcionar tarjetas y realizar el cobro de estas RNR03 – El despachador sólo puede acceder al sistema dentro de su horario laboral RNHA04 - Si el estacionamiento de queda sin lugares, entonces se muestra el mensaje “Cupo lleno”. RNI01 – Si el despachador indica que se entrega una tarjeta, entonces no se puede entregar la misma tarjeta a otro cliente.
Incluye:	Ninguno
Extiende:	Ninguno

Caso de uso 2: Cobrar tarjeta

ID:	CU02
Nombre del CU:	<i>Cobrar tarjeta</i>
Responsable:	Daniel Mongeote Tlachy
Fecha de actualización:	23/05/2023
Descripción:	El cliente le entrega la tarjeta al despachador al salir del estacionamiento y éste hace el respectivo cobro.
Actor(es):	- Despachador - Cliente
Disparador:	El despachador selecciona “Cobrar tarjeta” en el menú principal.
Precondiciones:	PRE01. El despachador debe tener una sesión iniciada con su cuenta en el sistema.
Flujo Normal:	<ol style="list-style-type: none"> 1. El sistema muestra una pantalla con un campo para ingresar el código de la tarjeta que se quiere indicar el cobro. 2. El Despachador ingresa el código de la tarjeta que el Cliente usó y selecciona “Mostrar” (Ver FA2.1, FA2.2). 3. El sistema muestra en la misma pantalla los datos de la tarjeta, el estatus de la tarifa y el nombre del despachador. 4. El Sistema muestra el botón “Iniciar proceso de pago”. 5. El Despachador selecciona el botón “Iniciar proceso de pago”. 6. El Sistema muestra un espacio para indicar el método de pago, una multa y un botón “Confirmar pago” (Ver EX01). 7. El despachador asigna un método de pago y selecciona el botón “Confirmar pago” (Ver FA7.1, FA7.2, FA7.3). 8. El sistema valida el pago y actualiza el registro. 9. El sistema modifica la disponibilidad de las tarjetas en la base de datos. 10. Termina caso de uso.
Flujos Alternos:	<p>FA2.1. El despachador ingresa un código que no existe.</p> <ol style="list-style-type: none"> 1. El sistema muestra “Tarjeta no encontrada”. 2. Ir al paso 2 (Flujo normal). <p>FA2.2. El despachador ingresa un código que sí existe pero que no corresponde a ningún registro pendiente de pago.</p> <ol style="list-style-type: none"> 1. El sistema muestra “La tarjeta no tiene un pago pendiente”. 2. Ir al paso 2 (Flujo normal). <p>FA.7.1. El despachador asigna una multa al cliente.</p> <ol style="list-style-type: none"> 1. El despachador selecciona una multa que muestra el sistema. 2. El sistema agrega la multa al costo total que debe pagar el

	<p>cliente.</p> <ol style="list-style-type: none"> 3. Ir al paso 4 (Flujo normal). <p>FA7.2. El despachador cancela la asignación de una multa al cliente.</p> <ol style="list-style-type: none"> 1. El despachador selecciona el ícono de regresar. 2. El sistema cancela la asignación de la multa 3. Ir al paso 1 (Flujo normal). <p>FA7.3. El despachador indica que la tarjeta se perdió</p> <ol style="list-style-type: none"> 1. El despachador selecciona “Tarjeta perdida”. 2. El sistema modifica los datos del registro, cambia el estado de la tarjeta a “Perdido” y muestra “La tarjeta ha sido registrada como perdida y el registro de ha guardado”.
Excepciones:	<p>EX01. Hubo un error con la conexión a la base de datos</p> <ol style="list-style-type: none"> 1. El sistema muestra el mensaje “No se pudo conectar con la base de datos. Inténtelo de nuevo o hágalo más tarde”. 2. Termina caso de uso.
Postcondiciones:	<p>POST01 - El sistema ha validado el cobro.</p> <p>POST02 – El cliente ha entregado la tarjeta.</p>
Reglas de negocio:	RNHA01 – Si el cliente pierde la tarjeta, entonces se le cobrará una multa.
Incluye:	Ninguno
Extiende:	Ninguno

Caso de uso 3: Administrar costos

ID:	CU03
Nombre del CU:	<i>Administrar costos</i>
Responsable:	Jesús Jacob Montiel Salas
Fecha de actualización:	10/05/2023
Descripción:	El administrador modifica el precio de las tarifas de pago y multas utilizadas en el estacionamiento
Actor(es):	- Administrador
Disparador:	El administrador selecciona “Administrar costos” en el menú principal
Precondiciones:	PRE01. El administrador debe tener una sesión iniciada con su cuenta en el sistema.
Flujo Normal:	<ol style="list-style-type: none"> 1. El sistema muestra una pantalla con los precios de tarifas y multas actuales del sistema divididos en secciones. 2. El administrador selecciona una tarifa de pago del apartado “Precio de multas” y selecciona “Modificar” (Ver FA2.1, FA2.2, FA2.3, FA2.4) 3. El sistema muestra la tarifa seleccionada con campos para modificarla 4. El administrador modifica los campos y selecciona “Confirmar” (Ver FA4.1, FA4.2) 5. El sistema actualiza la tarifa y muestra el mensaje “La tarifa ha sido modificada exitosamente” 6. El administrador selecciona una tarifa de pago del apartado

	<p>“Precio por horas para vehículos” y selecciona “Modificar” (Ver FA2.4)</p> <ol style="list-style-type: none"> 5. El sistema muestra la tarifa seleccionada con campos para modificarla 7. El administrador modifica los campos y selecciona “Confirmar” (Ver FA4.1, FA4.2) 8. El sistema actualiza la tarifa y muestra el mensaje “La tarifa ha sido modificada exitosamente” 9. El administrador selecciona una tarifa de pago del apartado “Precio por horas para motos” y selecciona “Modificar” (Ver FA2.4) 10. El sistema muestra la tarifa seleccionada con campos para modificarla 11. El administrador modifica los campos y selecciona “Confirmar” (Ver FA4.1, FA4.2) 12. El sistema actualiza la tarifa y muestra el mensaje “La tarifa ha sido modificada exitosamente”
Flujos Alternos:	<p>FA2.1. El administrador interrumpe el proceso de modificación de precios.</p> <ol style="list-style-type: none"> 1. El administrador selecciona Regresar. 2. El sistema regresa al administrador al menú principal. 3. Termina caso de uso. <p>FA2.2. El administrador elimina una tarifa de los registros.</p> <ol style="list-style-type: none"> 1. El administrador selecciona una tarifa y el botón “Eliminar”. (Ver FA2.2.1) 2. El sistema actualiza los registros y muestra el mensaje “La tarifa ha sido eliminada correctamente” <p>FA2.2.1. El administrador no selecciona una tarifa a eliminar.</p> <ol style="list-style-type: none"> 1. El administrador selecciona el botón “Eliminar”. 2. El sistema muestra el mensaje “Selecciona una tarifa para su eliminación” 3. Termina caso de uso <p>FA2.3. El administrador selecciona una tarifa de pago del apartado “Precio de multas” y selecciona “Añadir”</p> <ol style="list-style-type: none"> 1. El sistema muestra un formulario con los campos para registrar una nueva tarifa. 2. El administrador completa los campos y selecciona “Confirmar” (Ver FA4.1, FA4.2) 3. El sistema registra la nueva tarifa y muestra el mensaje “La tarifa fue añadida correctamente” 4. Termina caso de uso <p>FA2.4. El administrador no selecciona una tarifa a modificar</p> <ol style="list-style-type: none"> 1. El sistema muestra el mensaje “Selecciona una multa para su edición” 2. Termina caso de uso <p>FA4.1. El sistema detecta campos vacíos.</p> <ol style="list-style-type: none"> 1. El sistema muestra el mensaje: “Error. No puede haber

	<p>campos vacíos. Complételos para continuar.”</p> <ol style="list-style-type: none"> 2. Termina caso de uso. <p>FA4.1. El sistema detecta que se ha detectado un campo inválido en precio.</p> <ol style="list-style-type: none"> 11. El sistema muestra el mensaje: “Error. No es un formato de precio valido” 12. Termina caso de uso. <p>FA4.2. El administrador selecciona cancelar.</p> <ol style="list-style-type: none"> 1. El administrador selecciona “Cancelar” 2. El sistema le redirige a la lista de tarifas. 3. Termina caso de uso.
Excepciones:	<p>EX01. Hubo un error con la conexión a la base de datos.</p> <ol style="list-style-type: none"> 1. El sistema muestra el mensaje “No se pudo conectar con la base de datos. Inténtelo de nuevo o hágalo más tarde”. 2. Termina caso de uso.
Postcondiciones:	POST01 - El sistema actualiza los precios modificados en la base de datos para su posterior uso.
Reglas de negocio:	<p>RNR02 – El administrador es el único que puede administrar los usuarios, las tarjetas y los costos.</p> <p>RNHA01 – Si el cliente pierde la tarjeta, entonces se le cobrará una multa.</p> <p>RNHA03 - Si el cliente se queda después de la hora de cierre, entonces se le cobrará una multa.</p> <p>RNH05 – Los costos de las tarifas se pueden modificar por el administrador.</p>
Incluye:	Ninguno
Extiende:	Ninguno

Caso de uso 4: Administrar registros y tarjetas

ID:	CU04
Nombre del CU:	<i>Administrar registros y tarjetas</i>
Responsable:	Emilio Alejandro Rodríguez Pérez
Fecha de actualización:	11/05/2023
Descripción:	El Administrador cambia los estados de los registros y de las tarjetas
Actor(es):	- Administrador
Disparador:	El Administrador selecciona “Administrar registros y tarjetas” del menú principal.
Precondiciones:	PRE01. El administrador debe tener una sesión iniciada con su cuenta en el sistema.
Flujo Normal:	<ol style="list-style-type: none"> 1. El Sistema muestra la pantalla de “Administrar registros” y todos los registros de las tarjetas expedidas durante el día, junto con el detalle de sus descripciones como: “ID de tarjeta”, “número de cajón”, “fecha y hora de entrada”, “fecha y hora de salida”, “tiempo transcurrido”, “tipo de vehículo”, “tarifa asignada”, “tipo de método de pago” y “estatus de la tarifa”, junto con el botón “Guardar cambios”.

	<ol style="list-style-type: none"> 2. El administrador cambia el campo de estado de un registro y selecciona “Guardar cambios”. (Ver FA2.1) 3. El sistema guarda los cambios y muestra el mensaje “Registros modificados” (Ver EX01, FA3.1). 4. El administrador selecciona “Administrar tarjetas”. 5. El sistema muestra la pantalla de “Administrar tarjetas” y todas las tarjetas registradas en el sistema, cada una con su código, el estado actual de la tarjeta, el cajón y nivel al que corresponden. 6. El administrador selecciona el buscador, ingresa el código de la tarjeta y selecciona “Buscar”. 7. El sistema muestra las tarjetas que corresponden a la búsqueda (Ver EX01). 8. El Administrador cambia el estado de la tarjeta que se mostró y selecciona el botón “Guardar cambios”. 9. El sistema guarda los cambios y muestra el mensaje “Tarjetas modificadas” (Ver EX01).
Flujos Alternos:	<p>FA2.1. El Administrador selecciona otro día para ver los registros.</p> <ol style="list-style-type: none"> 1. El Administrador selecciona un día diferente para ver los registros. 2. El sistema muestra los registros con su información para ese día. 3. Ir al paso 2 (Flujo normal). <p>FA3.1. El Administrador se regresa al menú principal.</p> <ol style="list-style-type: none"> 1. El Administrador selecciona la opción de regresar. 2. El Sistema regresa al Administrador al menú principal y no guarda los cambios. 3. Termina caso de uso.
Excepciones:	<p>EX01. Hubo un error con la conexión a la base de datos</p> <ol style="list-style-type: none"> 1. El sistema muestra el mensaje “No se pudo conectar con la base de datos. Inténtelo de nuevo o hágalo más tarde”. 2. Termina caso de uso.
Postcondiciones:	POST01 - El sistema actualiza los estados de las tarjetas y de los registros.
Reglas de negocio:	RNH02 - Cada tarjeta corresponde a un cajón en específico. RNH03 - El administrador repone las tarjetas perdidas. RNHA01 – Si el cliente pierde la tarjeta, entonces se le cobrará una multa. RNHA02 - Si el cliente se estaciona en un cajón que no le corresponde, entonces se le cobrará una multa. RNHA03 - Si el cliente se queda después de la hora de cierre, entonces se le cobrará una multa.
Incluye:	Ninguno
Extiende:	Ninguno

Caso de uso 5: Generar reportes

ID:	CU05
Nombre del CU:	Generar reportes
Responsable:	Daniel Mongeote Tlachy
Fecha de	11/05/2023

actualización:	
Descripción:	El Administrador genera un historial de todos los registros de las tarjetas expedidas durante el día.
Actor(es):	- Administrador
Disparador:	El Administrador selecciona “Generar reportes” del menú principal.
Precondiciones:	PRE01. El administrador debe tener una sesión iniciada con su cuenta en el sistema.
Flujo Normal:	<ol style="list-style-type: none"> 1. El Sistema muestra el botón “Generar reporte” junto con el filtro para cambiar el día de trabajo laboral. 2. El Administrador no cambia el día para generar el reporte diario (Ver FA2.1). 3. El Administrador selecciona la opción “Generar reporte” (Ver FA4.1, FA4.2). 4. El Sistema genera el archivo del reporte y lo descarga en formato PDF (Ver EX01). 5. El Sistema muestra el mensaje “La descarga se ha completado correctamente”. 6. Termina caso de uso.
Flujos Alternos:	<p>FA2.1. El Administrador cambia el día de trabajo laboral.</p> <ol style="list-style-type: none"> 1. El Sistema actualiza los registros en base al día seleccionado. 2. Ir al paso 3. <p>FA4.1. El Administrador se regresa al menú principal.</p> <ol style="list-style-type: none"> 1. El Administrador selecciona la opción de regresar. 2. El Sistema regresa al Administrador al menú principal. 3. Termina caso de uso.
Excepciones:	<p>EX01. Hubo un error con la conexión a la base de datos</p> <ol style="list-style-type: none"> 1. El sistema muestra el mensaje “No se pudo conectar con la base de datos. Inténtelo de nuevo o hágalo más tarde”. 2. Termina caso de uso.
Postcondiciones:	POST01 - El sistema descarga el reporte en formato PDF correctamente.
Reglas de negocio:	<p>RNH02 - Cada tarjeta corresponde a un cajón en específico.</p> <p>RNH03 - El administrador repone las tarjetas perdidas.</p> <p>GenRepor.RQF01. El sistema debe permitir al administrador visualizar los registros de las tarjetas.</p> <p>GenRepor.RQF02. El sistema debe permitir al administrador modificar los estatus de los registros de las tarjetas.</p> <p>GenRepor.RQF03. El sistema debe permitir al administrador cambiar el día de generación de reportes.</p> <p>GenRepor.RQF04. El sistema debe permitir generar reportes de las tarjetas en formato PDF.</p> <p>GenRepor.RQF05. El sistema debe permitir al administrador descargar reportes de las tarjetas en formato PDF.</p>
Incluye:	Ninguno
Extiende:	Ninguno

Caso de uso 6: Crear usuario

ID:	CU06
Nombre del CU:	Crear usuario
Responsable:	Sulem Martínez Aguilar

Fecha de actualización:	11/05/2023
Descripción:	El administrador registra a un usuario nuevo en el sistema.
Actor(es):	- Administrador
Disparador:	El administrador selecciona “Crear usuario” en el apartado “Administración Usuarios”
Precondiciones:	PRE01. El administrador debe tener una sesión iniciada con su cuenta en el sistema.
Flujo Normal:	<ol style="list-style-type: none"> 1. El sistema muestra un formulario para ingresar el nombre, usuario, contraseña y tipo de usuario. 2. El administrador ingresa los datos especificados y selecciona de tipo “Despachador” (Ver FA2.1). 3. El sistema muestra un espacio para ingresar la jornada laboral en el que el despachador trabajará. 4. El administrador ingresa el horario en el que podrá trabajar el despachador. 5. El administrador selecciona “Guardar usuario”. (Ver FA5.1) 6. El sistema guarda el usuario en la base de datos, redirige al administrador al apartado “Administración usuarios” y muestra el mensaje “Usuario creado” (Ver FA6.1, EX01). 7. Termina caso de uso.
Flujos Alternos:	<p>FA1.1. El administrador registra a un usuario de tipo “Administrador”</p> <ol style="list-style-type: none"> 1. El administrador ingresa los datos especificados y selecciona de tipo “Administrador”. 2. Ir al paso 5 (Flujo normal). <p>FA5.1. El administrador se regresa antes de confirmar la creación del usuario.</p> <ol style="list-style-type: none"> 1. El administrador selecciona la opción de regresar. 2. El sistema redirige al administrador al apartado de “Administración de Usuarios”. 3. Termina caso de uso. <p>FA6.1. El administrador deja campos vacíos.</p> <ol style="list-style-type: none"> 1. El sistema muestra “No se pueden dejar campos vacíos”. 2. Ir al paso 2 (Flujo normal).
Excepciones:	<p>EX01. Hubo un error con la conexión a la base de datos</p> <ol style="list-style-type: none"> 1. El sistema muestra el mensaje “No se pudo conectar con la base de datos. Inténtelo de nuevo o hágalo más tarde”. 2. Termina caso de uso.
Postcondiciones:	<p>POST01 - El sistema guarda el usuario en la base de datos</p> <p>POST02 – El sistema redirige al administrador al apartado “Administración usuarios” y muestra el mensaje “Usuario creado”.</p>
Reglas de negocio:	RNR02 – El administrador es el único que puede administrar los usuarios, las tarjetas y los costos.
Incluye:	Ninguno
Extiende:	Ninguno

Caso de uso 7: Modificar usuario

ID:	CU06
Nombre del CU:	Modificar usuario
Responsable:	Sulem Martínez Aguilar

Fecha de actualización:	11/05/2023
Descripción:	El administrador consulta la lista de usuarios en el apartado “Administración Usuarios” para modificar un usuario existente.
Actor(es):	- Administrador
Disparador:	El administrador selecciona “Administración Usuarios” en el menú principal.
Precondiciones:	PRE01. El administrador debe tener una sesión iniciada con su cuenta en el sistema.
Flujo Normal:	<ol style="list-style-type: none"> 1. El sistema muestra la lista de todos los usuarios creados en el sistema y junto de cada uno hay un botón de “Ver detalles” (Ver FA1.1, EX01). 2. El administrador selecciona “Ver detalles” en uno de los elementos de la lista. 3. El sistema muestra el detalle del usuario y un botón de “Modificar usuario” (Ver EX01). 4. El administrador selecciona “Modificar usuario”. 5. El sistema muestra los campos del usuario como editables. 6. El administrador modifica los campos requeridos y selecciona “Guardar cambios” (Ver FA6.1). 7. El sistema guarda los cambios, redirige al administrador al apartado “Administración Usuarios” y muestra el mensaje “Usuario modificado” (Ver FA7.1, EX01). 8. Termina caso de uso
Flujos Alternos:	<p>FA1.1. No hay usuarios registrados en el sistema.</p> <ol style="list-style-type: none"> 1. El sistema muestra una lista vacía. 2. Termina caso de uso <p>FA6.1. El administrador se regresa antes de confirmar la creación del usuario.</p> <ol style="list-style-type: none"> 1. El administrador selecciona la opción de regresar. 2. El sistema redirige al administrador al apartado de “Administración de Usuarios”. 3. Termina caso de uso. <p>FA7.1. El administrador deja campos vacíos.</p> <ol style="list-style-type: none"> 1. El sistema muestra “No se pueden dejar campos vacíos”. 2. Ir al paso 6 (Flujo normal).
Excepciones:	<p>EX01. Hubo un error con la conexión a la base de datos</p> <ol style="list-style-type: none"> 3. El sistema muestra el mensaje “No se pudo conectar con la base de datos. Inténtelo de nuevo o hágalo más tarde”. 4. Termina caso de uso.
Postcondiciones:	<p>POST01 - EI El sistema guarda los cambios, redirige al administrador al apartado “Administración Usuarios” y muestra el mensaje “Usuario modificado” (Ver FA7.1, EX01).</p>
Reglas de negocio:	RNR02 – El administrador es el único que puede administrar los usuarios, las tarjetas y los costos. RNR04 – Una vez que se crea su cuenta, no se puede cambiar su tipo de usuario.
Incluye:	Ninguno
Extiende:	Ninguno

Caso de uso 8: Iniciar Sesión

ID:	CU07
Nombre del CU:	<i>Iniciar Sesión</i>
Responsable:	Jesús Jacob Montiel Salas
Fecha de actualización:	11/05/2023
Descripción:	El administrador o despachador inician sesión con su usuario y contraseña para ingresar al sistema
Actor(es):	- Administrador o despachador
Disparador:	El administrador o despachador inicia el sistema de gestión de estacionamiento
Precondiciones:	PRE01. El administrador o despachador deben tener una cuenta registrada en el sistema.
Flujo Normal:	<ol style="list-style-type: none"> 1. El sistema muestra los campos para ingresar un usuario y una contraseña 2. El administrador o profesor ingresa su usuario y contraseña en los campos correspondientes. (Ver FA2.1 FA2.2) 3. El administrador o despachador selecciona la opción “Iniciar sesión” (Ver FA3.1, FA3.2) 4. Termina Caso de Uso
Flujos Alternos:	<p>FA2.1. El administrador o despachador deja un campo vacío.</p> <ol style="list-style-type: none"> 1. El administrador o despachador selecciona la opción “Iniciar sesión” 2. El sistema muestra el mensaje “No se pueden dejar campos vacíos”. 3. Termina Caso de Uso <p>FA2.2. El administrador o despachador ingresa un usuario que no existe en el sistema.</p> <ol style="list-style-type: none"> 1. El administrador o despachador selecciona la opción “Iniciar sesión” 2. El sistema muestra el mensaje “El usuario no está registrado”. 3. Termina Caso de Uso <p>FA3.1. El administrador ingresa al sistema en horario de mantenimiento.</p> <ol style="list-style-type: none"> 1. El administrador selecciona la opción “Iniciar sesión” 2. El sistema muestra el mensaje “No se puede acceder al sistema en horas de mantenimiento”. 3. Termina Caso de Uso <p>FA3.2. El despachador ingresa al sistema en horario fuera de su jornada laboral.</p> <ol style="list-style-type: none"> 1. El despachador selecciona la opción “Iniciar sesión” 2. El sistema muestra el mensaje “No se puede acceder al sistema fuera de jornada laboral.”. 3. Termina Caso de Uso
Excepciones:	<p>EX01. Hubo un error con la conexión a la base de datos</p> <ol style="list-style-type: none"> 1. El sistema muestra el mensaje “No se pudo conectar con la base de datos. Inténtelo de nuevo o hágalo más tarde”. 2. Termina caso de uso.
Postcondiciones:	<p>POST01 - El sistema valida como correctas las credenciales del usuario</p> <p>POST02 - El sistema muestra al usuario su menú principal</p>
Reglas de negocio:	RNR03 – El despachador sólo puede acceder al sistema dentro de su jornada laboral.

	RNR03 – El administrador no puede acceder al sistema en horario de mantenimiento RNH01 - El horario laboral del estacionamiento es de 7am a 8pm para despachadores.
Incluye:	Ninguno
Extiende:	Ninguno

8. Diseño

Diagrama Entidad-Relación

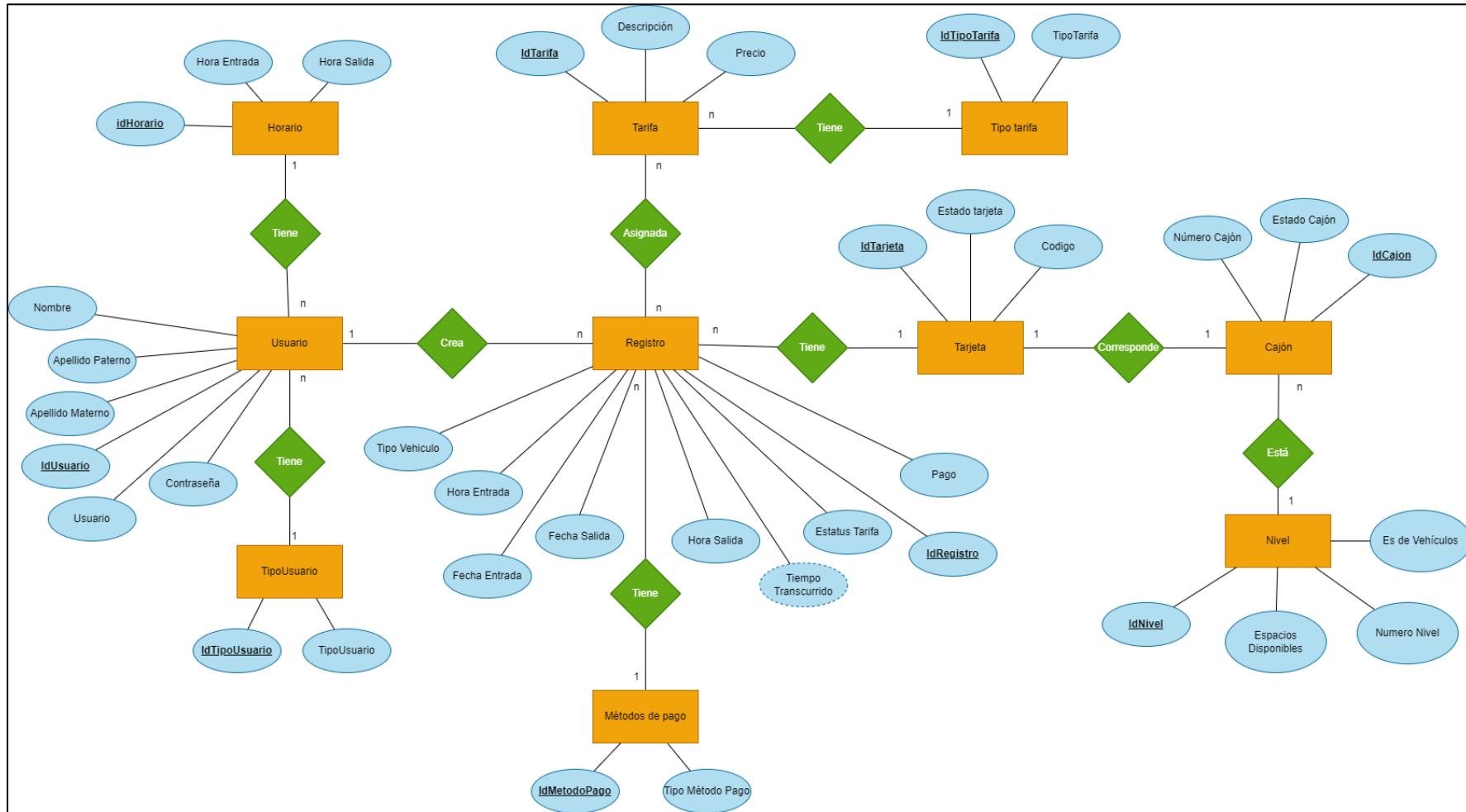


Ilustración 8.1 Modelo Entidad Relación

Diagrama Relacional

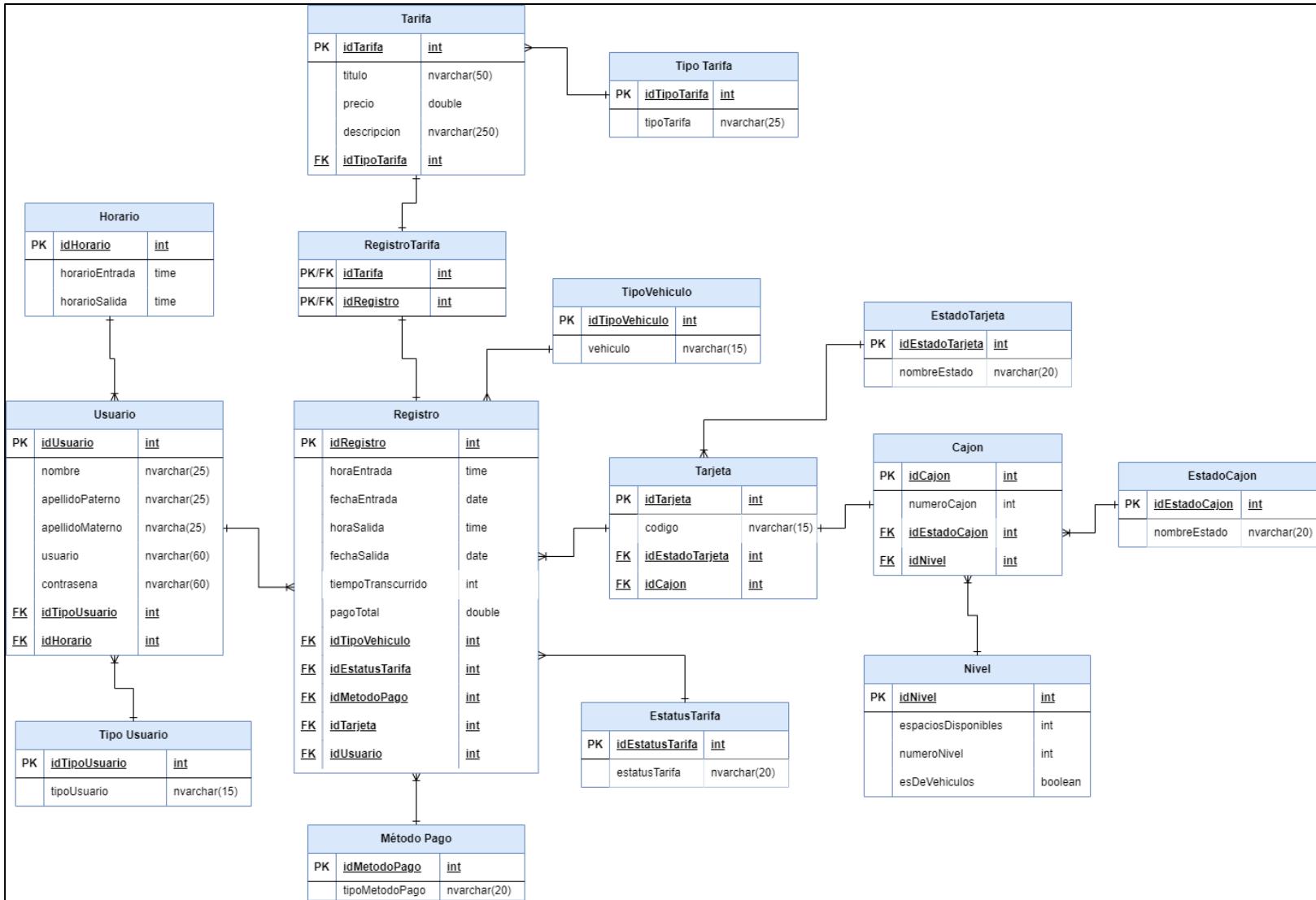


Ilustración 8.2 Diagrama relacional

Diagrama de Clases

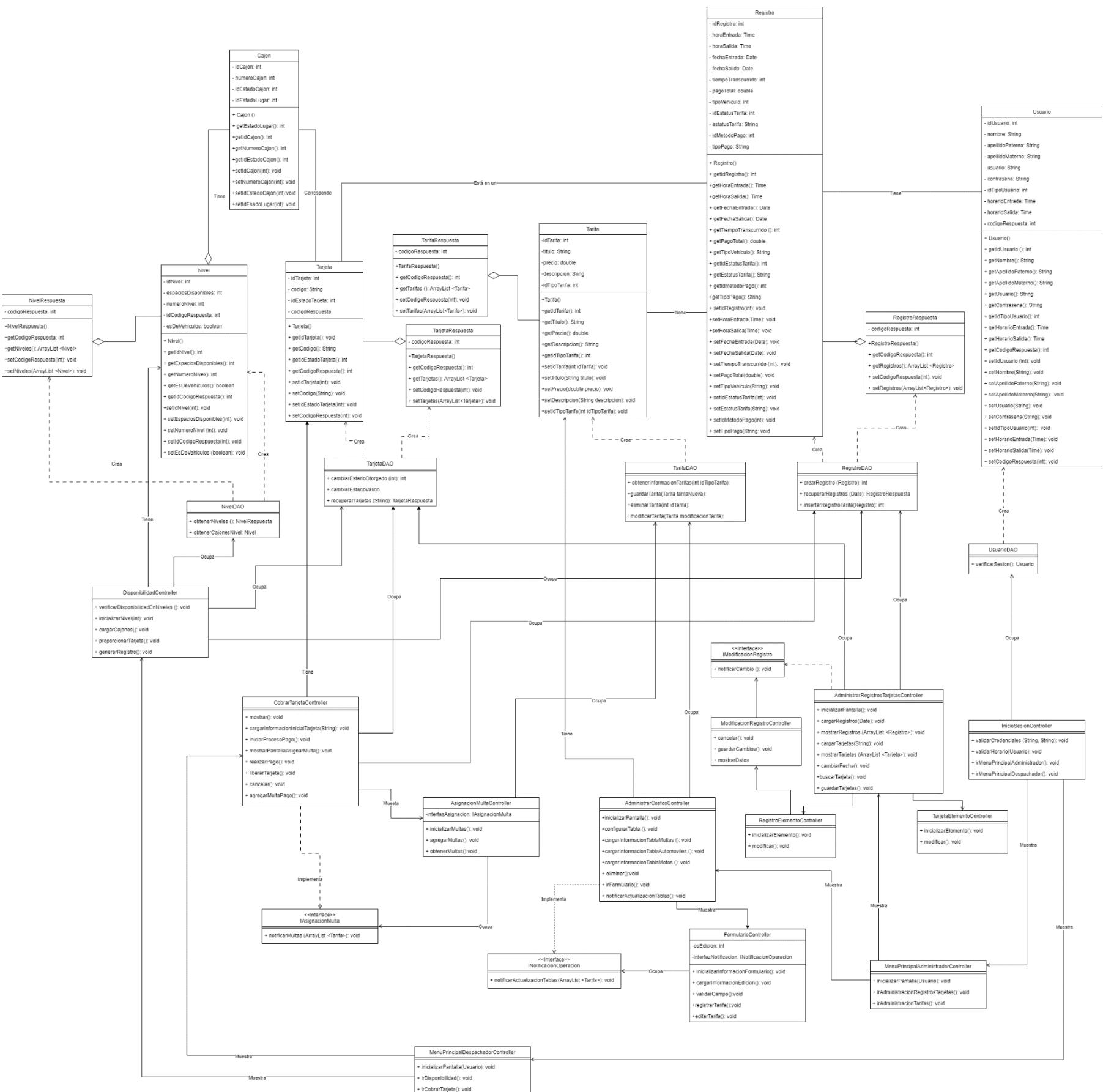


Ilustración 8.3 Modelo Clases

9. Implementación

Se realizó una matriz de pruebas en el documento siguiente:

[MatrizPrueba.xlsx](#)

Pruebas unitarias

Se realizaron las pruebas unitarias con la biblioteca JUnit 4.13 para comprobar el funcionamiento correcto de los DAO creados para el sistema:

NivelDAO

Obtener Niveles del Estacionamiento:

The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and JavaFXSistemaEstacionamientoJETS - Apache NetBeans IDE 13. The search bar says "Search (Ctrl+I)". The main window has tabs for Start Page, JavaFXSistemaEstacionamientoJETS.java, NivelDAOIT.java, and NivelDAO.java. The NivelDAO.java tab is active, displaying Java code for testing the NivelDAO class. The code includes annotations @After and @Test, and methods for testing the obtenerNiveles and obtenerCajonesNivel methods. Below the code editor is the Output - JavaFXSistemaEstacionamientoJETS (test-single-method) window, which shows "Tests passed: 100.00%" and "Obtener Niveles del Estacionamiento". The status bar at the bottom right shows the date and time: 41:54, INS Windows (CRLF), 11:56 p.m., 02/06/2023.

```
31
32     @After
33     public void tearDown() {
34         }
35
36     /**
37      * Test of obtenerNiveles method, of class NivelDAO.
38     */
39     @Test
40     public void testObtenerNiveles() {
41         System.out.println("Obtener Niveles del Estacionamiento");
42         NivelRespuesta expResult = new NivelRespuesta();
43         expResult.setCodigoRespuesta(Constantes.OPERACION_EXITOGA);
44         NivelRespuesta result = NivelDAO.obtenerNiveles();
45         assertEquals(expResult.getCodigoRespuesta(), result.getCodigoRespuesta());
46     }
47
48     /**
49      * Test of obtenerCajonesNivel method, of class NivelDAO.
50     */
51     @Test
```

Obtener cajones de un nivel:

The screenshot shows the Apache NetBeans IDE interface, similar to the previous one. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and JavaFXSistemaEstacionamientoJETS - Apache NetBeans IDE 13. The search bar says "Search (Ctrl+I)". The main window has tabs for Start Page, JavaFXSistemaEstacionamientoJETS.java, NivelDAOIT.java, and NivelDAO.java. The NivelDAO.java tab is active, displaying Java code for testing the NivelDAO class. The code includes annotations @After and @Test, and methods for testing the obtenerNiveles and obtenerCajonesNivel methods. Below the code editor is the Output - JavaFXSistemaEstacionamientoJETS (test-single-method) window, which shows "Tests passed: 100.00%" and "Obtener Cajones de un Nivel del estacionamiento". The status bar at the bottom right shows the date and time: 52:32, INS Windows (CRLF), 11:58 p.m., 02/06/2023.

```
41
42     public void testObtenerNiveles() {
43         System.out.println("Obtener Niveles del Estacionamiento");
44         NivelRespuesta expResult = new NivelRespuesta();
45         expResult.setCodigoRespuesta(Constantes.OPERACION_EXITOSA);
46         NivelRespuesta result = NivelDAO.obtenerNiveles();
47         assertEquals(expResult.getCodigoRespuesta(), result.getCodigoRespuesta());
48     }
49
50     /**
51      * Test of obtenerCajonesNivel method, of class NivelDAO.
52     */
53     @Test
54     public void testObtenerCajonesNivel() {
55         System.out.println("Obtener Cajones de un Nivel del estacionamiento");
56         int numeroNivel = 1;
57         Nivel expResult = new Nivel();
58         expResult.setCodigoRespuesta(Constantes.OPERACION_EXITOSA);
59         Nivel result = NivelDAO.obtenerCajonesNivel(numeroNivel);
60         assertEquals(expResult.getCodigoRespuesta(), result.getCodigoRespuesta());
```

RegistroDAO

Crear registro con datos correctos:

The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and JavaFXSistemaEstacionamientoJETS - Apache NetBeans IDE 13. The title bar shows the current project name. The main window has several tabs open: JavaFXSistemaEstacionamientoJETS.java, NivelDAOIT.java, RegistroDAOIT.java (which is the active tab), XMLInicioSesionController.java, and FXMLDisponibilidadController.java. The Navigator panel on the left lists various Java files under the Source Packages section. The Source editor on the right contains the code for the test method `testCrearRegistro()`. The Output and Test Results panes at the bottom show a green bar indicating 100.00% tests passed, with the message "The test passed. (0.485 s)". The status bar at the bottom right shows the date and time as 12:19 a.m. 03/06/2023.

```
48     @Test
49     public void testCrearRegistro() {
50         System.out.println("Crear Registro");
51         Registro registroNuevo = new Registro();
52
53         Date hoy = new Date();
54         java.sql.Date fechaEntrada = new java.sql.Date(hoy.getTime());
55         Calendar calendarioInstancia = Calendar.getInstance();
56         java.sql.Time tiempoEntrada = new java.sql.Time(calendarioInstancia.getTime().getTime());
57
58         registroNuevo.setHoraEntrada(tiempoEntrada);
59         registroNuevo.setFechaEntrada(fechaEntrada);
60         registroNuevo.setIdTipoVehiculo(Constantes.VEHICULO);
61         registroNuevo.setIdStatusTarifa(Constantes.ESTATUS_TARIFA_EN_PROCESO);
62         registroNuevo.setIdTarjeta(1);
63         registroNuevo.setIdUsuario(1);
64         registroNuevo.setIdMetodoPago(Constantes.METODO_PAGO_PENDIENTE);
65         int expResult = Constantes.OPERACION_EXITOSA;
66         int result = RegistroDAO.createRegistro(registroNuevo);
67         assertEquals(expResult, result);
68     }
```

Crear registro con datos incorrectos:

This screenshot shows the same Apache NetBeans IDE setup as the previous one, but the test results are different. The Output and Test Results panes now indicate 0.00% tests passed, with a red bar and the message "No test passed, 1 test failed. (0.416 s)". A detailed error message is shown in the Test Results pane: "javafxsistemasestacionamientojets.modelo.dao.RegistroDAOIT Failed > testCrearRegistro Failed: Prueba fallida en crear un nuevo registro". The stack trace for this failure is also visible. The status bar at the bottom right shows the date and time as 12:32 a.m. 03/06/2023.

```
48     @Test
49     public void testCrearRegistro() {
50         System.out.println("Crear Registro");
51         Registro registroNuevo = new Registro();
52
53         Date hoy = new Date();
54         java.sql.Date fechaEntrada = new java.sql.Date(hoy.getTime());
55         Calendar calendarioInstancia = Calendar.getInstance();
56         java.sql.Time tiempoEntrada = new java.sql.Time(calendarioInstancia.getTime().getTime());
57
58         registroNuevo.setHoraEntrada(tiempoEntrada);
59         registroNuevo.setFechaEntrada(fechaEntrada);
60         registroNuevo.setIdTipoVehiculo(0);
61         registroNuevo.setIdStatusTarifa(0);
62         registroNuevo.setIdTarjeta(0);
63         registroNuevo.setIdUsuario(0);
64         registroNuevo.setIdMetodoPago(0);
65         int expResult = Constantes.OPERACION_EXITOSA;
66         int result = RegistroDAO.createRegistro(registroNuevo);
67         //assertEquals(expResult, result);
68         assertEquals("Prueba fallida en crear un nuevo registro", expResult == result);
```

Recuperar registro pendiente de pago:

```

84 /**
85 * Test of recuperarRegistroPendientePago method, of class RegistroDAO.
86 */
87 @Test
88 public void testRecuperarRegistroPendientePago() {
89     System.out.println("Recuperar un Registro Pendiente de Pago");
90     int idTarjeta = 17;
91     Registro expResult = new Registro();
92     expResult.setCodigoRespuesta(Constantes.OPERACION_EXITOSA);
93     Registro result = RegistroDAO.recuperarRegistroPendientePago(idTarjeta);
94     assertEquals(expResult.getCodigoRespuesta(), result.getCodigoRespuesta());
95 }
96 /**
97 * Test of modificarRegistro method, of class RegistroDAO.
98 */
99 @Test
100 public void testModificarRegistro() {
101     System.out.println("modificarRegistro");
102 }
103

```

Modificar registro con datos correctos:

```

100 /**
101 * Test of modificarRegistro method, of class RegistroDAO.
102 */
103 @Test
104 public void testModificarRegistro() {
105     System.out.println("Modificar registro");
106     Registro registroModificado = new Registro();
107     Date hoy = new Date();
108     java.sql.Date fechaEntrada = new java.sql.Date(hoy.getTime());
109     Calendar calendarioInstancia = Calendar.getInstance();
110     java.sql.Time tiempoEntrada = new java.sql.Time(calendarioInstancia.getTime().getTime());
111     registroModificado.setIdRegistro(17);
112     registroModificado.setHoraSalida(tiempoEntrada);
113     registroModificado.setFechaSalida(fechaEntrada);
114     registroModificado.setTiempoTranscurrido(10);
115     registroModificado.setPagoTotal(100);
116     registroModificado.setIDMetodoPago(1);
117     int expResult = Constantes.OPERACION_EXITOSA;
118     int result = RegistroDAO.modificarRegistro(registroModificado);
119     assertEquals(expResult, result);
120 }
121

```

Modificar registro con datos incorrectos:

The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, and Help. The title bar says "JavaFXSistemaEstacionamientoJETS - Apache NetBeans IDE 13". The main window has a Navigator on the left listing various Java files and packages. The Source tab is selected, displaying Java code for a test method named "testModificarRegistro". The code initializes a "Registro" object with current date and time, sets an entry time, and a duration of 10 minutes. It then attempts to modify the registration with a total page count of 100 and a payment method ID of 1. The expected result is set to "OPERACION_EXITOSA". The code then asserts that the result matches the expected value and that the test failed if it does not. Below the source code is an Output window titled "Output - JavaFXSistemaEstacionamientoJETS (test-single)" which shows a green status bar indicating "Tests passed: 25.00%". The Test Results window shows one test passed and three tests failed, with detailed error messages for each failure.

```
101     @Test
102     public void testModificarRegistro() {
103         System.out.println("Modificar registro");
104         Registro registroModificado = new Registro();
105         Date hoy = new Date();
106         java.sql.Date fechaEntrada = new java.sql.Date(hoy.getTime());
107         Calendar calendarioInstancia = Calendar.getInstance();
108         java.sql.Time tiempoEntrada = new java.sql.Time(calendarioInstancia.getTime().getTime());
109         registroModificado.setIdRegistro(0);
110         registroModificado.setHoraSalida(tiempoEntrada);
111         registroModificado.setFechaSalida(fechaEntrada);
112         registroModificado.setTiempoTranscurrido(10);
113         registroModificado.setPagoTotal(100);
114         registroModificado.setIdMetodoPago(1);
115         int expResult = Constantes.OPERACION_EXITOSA;
116         int result = RegistroDAO.modificarRegistro(registroModificado);
117         //assertEquals(expResult, result);
118         assertTrue("Prueba fallida en modificar un registro",expResult == result);
119     }
120 }
```

Output - JavaFXSistemaEstacionamientoJETS (test-single) × Test Results ×

JavaFXSistemaEstacionamientoJETS

Tests passed: 25.00%

1 test passed, 3 tests failed. (0.526 s)

JavaSistemaEstacionamientoJETS.modelo.dao.RegistroDAOIT Failed

testModificarRegistro Failed: Prueba fallida en modificar un registro

testRecuperarRegistros Failed: expected:<null> but was:<javafx.sistemaestacionamiento.J

testCrearRegistro Failed: Prueba fallida en crear un nuevo registro

at org.junit.runners.ParentRunner\$3.evaluate(ParentRunner.java:306)
at org.junit.runners.ParentRunner.run(ParentRunner.java:413)
at junit.framework.JUnit4TestAdapter.run(JUnit4TestAdapter.java:50)
at org.apache.tools.ant.taskdefs.optional.junit.JUnitTestRunner.it
at org.apache.tools.ant.taskdefs.optional.junit.JUnitTestRunner.la
at org.apache.tools.ant.taskdefs.optional.junit.JUnitTestRunner.ma

102:31 INS 12:33 a.m. 03/06/2023

RegistroTarifaDAO

Insertar un registro tarifa con datos correctos:

The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, and Help. The title bar says "JavaFXSistemaEstacionamientoJETS - Apache NetBeans IDE 13". The main window has a Navigator on the left listing various Java files and packages. The Source tab is selected, displaying Java code for a test method named "testInsertarRegistroTarifa". The code creates a Tarifa object with ID 1, adds it to a list of Tarifas, and then inserts it into a Registro object with ID 17. The expected result is set to "OPERACION_EXITOSA". The code then asserts that the result matches the expected value. Below the source code is an Output window titled "Output - JavaFXSistemaEstacionamientoJETS (test-single-method)" which shows a green status bar indicating "Tests passed: 100.00%". The Test Results window shows one test passed with a duration of 0.367 seconds.

```
42     /**
43      * Test of insertarRegistroTarifa method, of class RegistroTarifaDAO.
44      */
45     @Test
46     public void testInsertarRegistroTarifa() {
47         System.out.println("insertarRegistroTarifa");
48         Tarifa tarifa = new Tarifa();
49         tarifa.setIdTarifa(1);
50         ArrayList<Tarifa> tarifas = new ArrayList();
51         tarifas.add(tarifa);
52         Registro registro = new Registro();
53         registro.setIdRegistro(17);
54         registro.setTarifas(tarifas);
55         int expResult = Constantes.OPERACION_EXITOSA;
56         int result = RegistroTarifaDAO.insertarRegistroTarifa(registro);
57         assertEquals(expResult, result);
58     }
59 }
60
61 }
```

Output - JavaFXSistemaEstacionamientoJETS (test-single-method) × Test Results ×

JavaFXSistemaEstacionamientoJETS

Tests passed: 100.00%

The test passed. (0.367 s)

insertarRegistroTarifa

47:33 INS Windows (CRLF) 12:38 a.m. 03/06/2023

Insertar un registro tarifa con datos incorrectos:

The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and JavaFXSistemaEstacionamientoJETS - Apache NetBeans IDE 13. The title bar shows the project name and version. The left sidebar has a Projects view with several Java files listed under JavaFXSistemaEstacionamientoJETS. The main workspace displays a Java file named RegistroTarifaDAOIT.java with the following code:

```
43 /**
44 * Test of insertarRegistroTarifa method, of class RegistroTarifaDAO.
45 */
46 @Test
47 public void testInsertarRegistroTarifa() {
48     System.out.println("insertarRegistroTarifa");
49     Tarifa tarifa = new Tarifa();
50     tarifa.setIdTarifa(1);
51     ArrayList<Tarifa> tarifas = new ArrayList();
52     tarifas.add(tarifa);
53     Registro registro = new Registro();
54     registro.setIdRegistro(0);
55     registro.setTarifas(tarifas);
56     int expResult = Constantes.OPERACION_EXITOSA;
57     int result = RegistroTarifaDAO.insertarRegistroTarifa(registro);
58     //assertEquals(expResult, result);
59     assertEquals("Prueba fallida en insertar una tarifa a un registro", expResult == result);
60 }
61
62 }
```

Below the code editor is an Output window titled "Output - JavaFXSistemaEstacionamientoJETS (test-single-method)" showing the results of the test run:

```
Tests passed: 0.00 %
No test passed, 1 test failed. (0.395 s)
java.sistemaestacionamientojets.modelo.dao.RegistroTarifaDAOIT Failed
> testInsertarRegistroTarifa Failed: Prueba fallida en insertar una tarifa a un registro
```

The status bar at the bottom shows the build status as "Build of JavaFXSistemaEstacionamientoJETS (test-single-method) failed.", the time as 12:42 a.m. 03/06/2023, and the operating system as INS Windows (CRLF).

TarifaDAO

Obtener información tarifa:

The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and JavaFXSistemaEstacionamientoJETS - Apache NetBeans IDE 13. The title bar shows the project name and version. The left sidebar has a Projects view with several Java files listed under JavaFXSistemaEstacionamientoJETS. The main workspace displays a Java file named TarifaDAO.java with the following code:

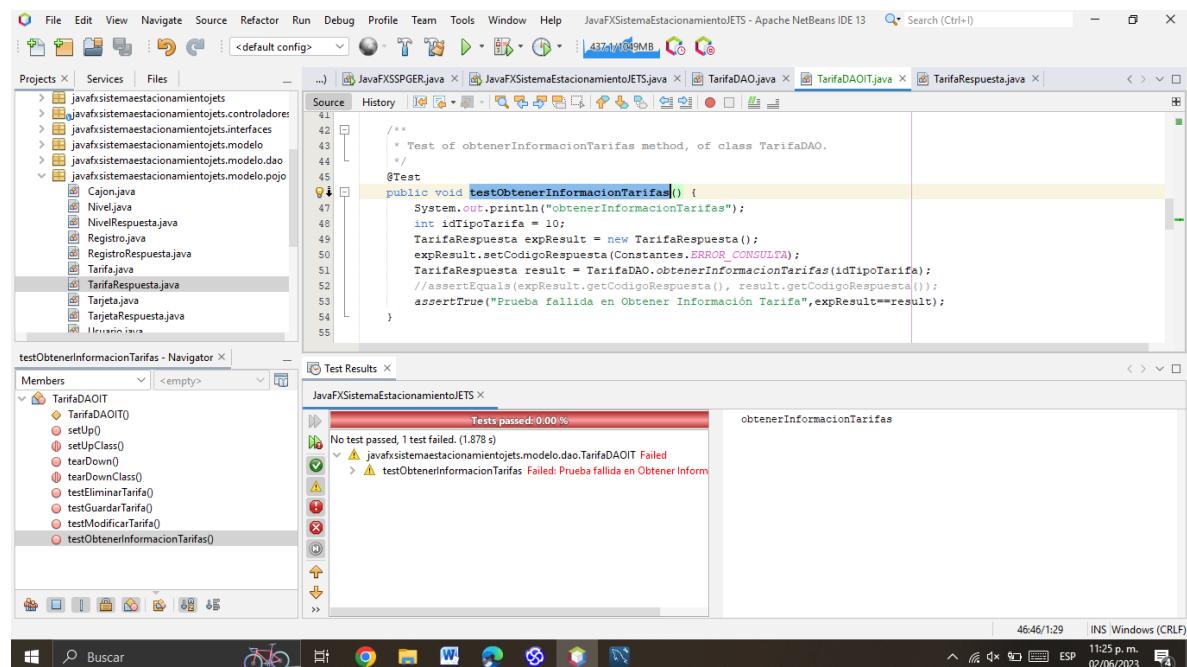
```
41 /**
42 * Test of obtenerInformacionTarifas method, of class TarifaDAO.
43 */
44 @Test
45 public void testObtenerInformacionTarifas() {
46     System.out.println("obtenerInformacionTarifas");
47     int idIipoTarifa = 1;
48     TarifaRespuesta expResult = new TarifaRespuesta();
49     expResult.setCodigoRespuesta(Constantes.OPERACION_EXITOSA);
50     TarifaRespuesta result = TarifaDAO.obtenerInformacionTarifas(idIipoTarifa);
51     assertEquals(expResult.getCodigoRespuesta(), result.getCodigoRespuesta());
52     //assertTrue("Prueba fallida en Obtener Información Tarifa", expResult==result);
53 }
```

Below the code editor is a Test Results window titled "Test Results" showing the results of the test run:

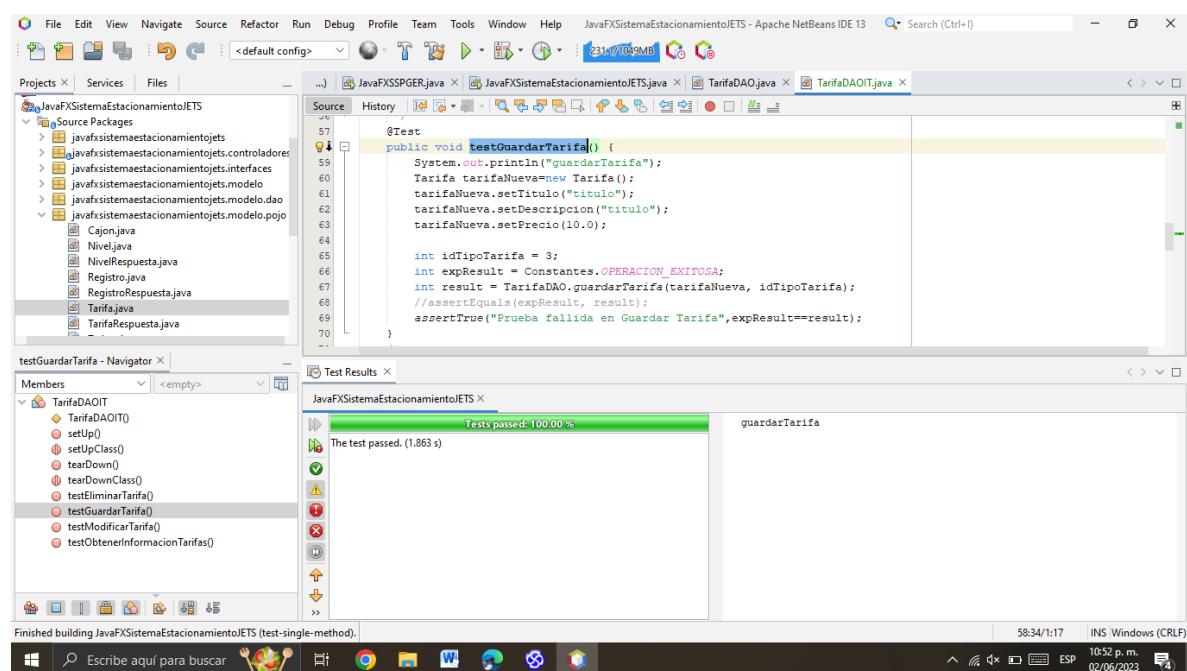
```
Tests passed: 100.00 %
The test passed. (1.771 s)
java.sistemaestacionamientojets.modelo.dao.TarifaDAOIT passed
testObtenerInformacionTarifas passed (1.465 s)
```

The status bar at the bottom shows the build status as "Finished building JavaFXSistemaEstacionamientoJETS (test-single-method).", the time as 11:23 p.m. 02/06/2023, and the operating system as INS Windows (CRLF).

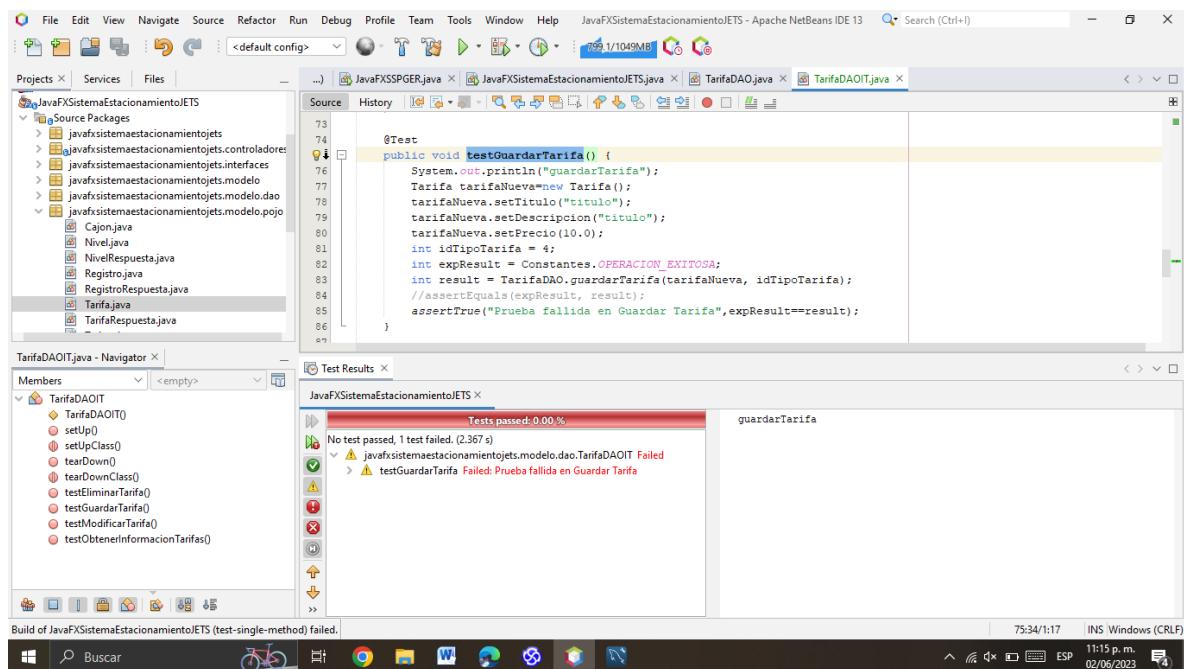
Obtener información tarifa con un tipo de tarifa que no existe:



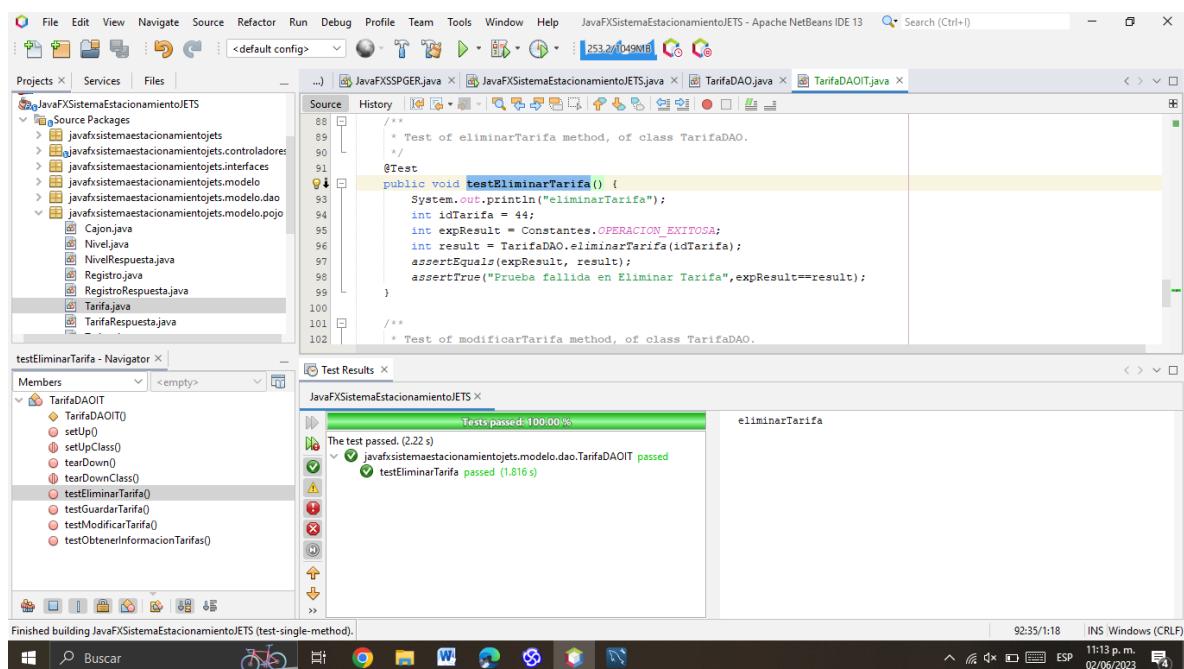
Guardar una tarifa nueva:



Guardar tarifa con un Tipo de Tarifa que no existe:



Eliminar tarifa:



Eliminar tarifa que no existe:

The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and JavaFXSistemaEstacionamientoJETS - Apache NetBeans IDE 13. The bottom status bar shows memory usage (882.97/1049MB), date (02/06/2023), and time (11:12 p.m.).

The left sidebar displays the project structure under "Source Packages". The "TarifaDAOIT" package is selected, showing test methods like setUp(), tearDown(), tearDownClass(), testEliminarTarifa(), testGuardarTarifa(), testModificarTarifa(), and testObtenerInformacionTarifas().

The main editor window shows Java code for the "testEliminarTarifa" method in the "TarifaDAOIT" class. The code attempts to delete a tariff with ID 2, but since it doesn't exist, it fails with an exception.

```
88 /**
89 * Test of eliminarTarifa method, of class TarifaDAO.
90 */
91 @Test
92 public void testEliminarTarifa() {
93     System.out.println("eliminarTarifa");
94     int idTarifa = 2;
95     int expResult = Constantes.OPERACION_EXITOSA;
96     int result = TarifaDAO.eliminarTarifa(idTarifa);
97     //assertEquals(expResult, result);
98     assertTrue("Prueba fallida en Eliminar Tarifa", expResult==result);
99 }
100 /**
101 * Test of modificarTarifa method, of class TarifaDAO.
```

The "Test Results" panel shows a red bar indicating "Tests passed: 0.00%". A single test case, "testEliminarTarifa", is listed as failed with the message "Prueba fallida en Eliminar Tarifa".

Modificar tarifa existente:

The screenshot shows the Apache NetBeans IDE interface, identical to the previous one but with different code in the editor.

The left sidebar displays the project structure under "Source Packages". The "TarifaDAOIT" package is selected, showing test methods like setUp(), tearDown(), tearDownClass(), testEliminarTarifa(), testGuardarTarifa(), testModificarTarifa(), and testObtenerInformacionTarifas().

The main editor window shows Java code for the "testModificarTarifa" method in the "TarifaDAOIT" class. The code creates a new tariff object, sets its ID to 43, and then updates its title to "titulo2", description to "descripcion", and price to 100.0.

```
89 /**
90 * TEST OR MODIFICAR TARIFFE METODO, OF CLASS TARIFADAO.
91 */
92 @Test
93 public void testModificarTarifa() {
94     System.out.println("modificarTarifa");
95     Tarifa modificaciónTarifa=new Tarifa();
96     modificaciónTarifa.setIdtarifa(43);
97     modificaciónTarifa.setTitulo("titulo2");
98     modificaciónTarifa.setDescripcion("descripcion");
99     modificaciónTarifa.setPrecio(100.0);
100    int expResult = Constantes.OPERACION_EXITOSA;
101    int result = TarifaDAO.modificarTarifa(modificaciónTarifa);
102    assertEquals(expResult, result);
103    assertTrue("Prueba fallida en Modificar Tarifa", expResult==result);
104 }
```

The "Test Results" panel shows a green bar indicating "Tests passed: 100.00%". All test cases under "javafxsistemaestacionamientojots.modelo.dao.TarifaDAOIT" are marked as passed.

Devolver el precio de una tarifa con base al tipo de tarifa y las horas totales:

```
96     }
97     }
98     */
99     /**
100    * Test of calcularPrecio method, of class TarifaDAO.
101   */
102  @Test
103  public void testCalcularPrecio() {
104      System.out.println("calcularPrecio");
105      int idTipoTarifa = Constantes.OPERACION_EXITOSA;
106      int horas = 10;
107      Tarifa expResult = new Tarifa();
108      expResult.setCodigoRespuesta(Constantes.OPERACION_EXITOSA);
109      Tarifa result = TarifaDAO.calcularPrecio(idTipoTarifa, horas);
110      assertEquals(expResult.getCodigoRespuesta(), result.getCodigoRespuesta());
111  }
112 }
113 }
114 }
```

Output - JavaFXSistemaEstacionamientoJETS (test-single-method) x Test Results x

JavaFXSistemaEstacionamientoJETS x

Tests passed: 100.00 %

The test passed. (0.382 s)

cambiarEstado

103:23 INS Windows (CRLF)

TarjetaDAO

Cambiar estado de una tarjeta con datos correctos:

```
37
38     @After
39     public void tearDown() {
40     }
41
42     /**
43      * Test of cambiarEstado method, of class TarjetaDAO.
44     */
45  @Test
46  public void testCambiarEstado() {
47      System.out.println("cambiarEstado");
48      Tarjeta tarjeta = new Tarjeta();
49      tarjeta.setIdTarjeta(1);
50      tarjeta.setIdEstadoTarjeta(2);
51      int expResult = Constantes.OPERACION_EXITOSA;
52      int result = TarjetaDAO.cambiarEstado(tarjeta);
53      assertEquals(expResult, result);
54  }
55 }
```

Output - JavaFXSistemaEstacionamientoJETS (test-single-method) x Test Results x

JavaFXSistemaEstacionamientoJETS x

Tests passed: 100.00 %

The test passed. (0.381 s)

cambiarEstado

46:28 INS Windows (CRLF)

Cambiar estado de una tarjeta con datos incorrectos:

The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and JavaFXSistemaEstacionamientoJETS - Apache NetBeans IDE 13. The toolbar has icons for file operations like Open, Save, and Print. The left sidebar (Navigator) lists projects: JavaFXConvertidorDivisas, JavaFXHello, JavaFXPruebaConLista, JavaFXPruebaSistemaEstacionamiento, and JavaFXSistemaEstacionamiento. Under JavaFXSistemaEstacionamiento, there are Source Packages (javafxsistemaestacionamiento, javafxsistemaestacionamiento, javafxsistemaestacionamiento, javafxsistemaestacionamiento, and TarjetaDAO.java) and Test Packages (javafxsistemaestacionamiento, TarjetaDAOIT.java). The main Source editor window displays Java code for testing the TarjetaDAO class. The code includes a test method `testCambiarEstado()` that creates a new `Tarjeta` object, sets its ID to 0, and then changes its state to 2. It then checks if the result matches the expected value from the `Constantes` class. The `Output - JavaFXSistemaEstacionamientoJETS (test-single-method)` window shows a red bar indicating "Tests passed: 0.00 %". Below it, a warning message says "No test passed, 1 test failed. (0.377 s)" and details a failure for the `testCambiarEstado` method: "Failed: Prueba fallida en cambiar el estado de una tarjeta". The status bar at the bottom shows the date and time: 12:46 a.m. 03/06/2023.

```
    /**
     * Test of cambiarEstado method, of class TarjetaDAO.
     */
    @Test
    public void testCambiarEstado() {
        System.out.println("cambiarEstado");
        Tarjeta tarjeta = new Tarjeta();
        tarjeta.setIdTarjeta(0);
        tarjeta.setEstadoTarjeta(2);
        int expResult = Constantes.OPERACION_EXITOSA;
        int result = TarjetaDAO.cambiarEstado(tarjeta);
        assertEquals(expResult, result);
        //assertEquals("Prueba fallida en cambiar el estado de una tarjeta",expResult == result);
    }

    /**
     * Test of recuperarTarjetas method, of class TarjetaDAO.
     */
}
```

Recuperar una tarjeta con base a su código:

The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and JavaFXSistemaEstacionamientoJETS - Apache NetBeans IDE 13. The toolbar has icons for file operations like Open, Save, and Print. The left sidebar (Navigator) lists projects: JavaFXConvertidorDivisas, JavaFXHello, JavaFXPruebaConLista, JavaFXPruebaSistemaEstacionamiento, and JavaFXSistemaEstacionamiento. Under JavaFXSistemaEstacionamiento, there are Source Packages (javafxsistemaestacionamiento, javafxsistemaestacionamiento, javafxsistemaestacionamiento, javafxsistemaestacionamiento, and TarjetaDAO.java) and Test Packages (javafxsistemaestacionamiento, TarjetaDAOIT.java). The main Source editor window displays Java code for testing the TarjetaDAO class. The code includes a test method `testRecuperarTarjeta()` that prints a message, sets a card code ("Nivel1cajon17"), creates an expected result object, sets its response code to successful, retrieves a card by its code, and asserts that the retrieved card's response code matches the expected one. The `Output - JavaFXSistemaEstacionamientoJETS (test-single-method)` window shows a green bar indicating "Tests passed: 100.00 %". Below it, a success message says "The test passed. (0.384 s)". The status bar at the bottom shows the date and time: 12:47 a.m. 03/06/2023.

```
    /**
     * TODO review the generated test code and remove the default call to fail.
     * fail("The test case is a prototype.");
    }

    /**
     * Test of recuperarTarjeta method, of class TarjetaDAO.
     */
    @Test
    public void testRecuperarTarjeta() {
        System.out.println("recuperarTarjeta");
        String codigo = "Nivel1cajon17";
        Tarjeta expResult = new Tarjeta();
        expResult.setCodigoRespuesta(Constantes.OPERACION_EXITOSA);
        Tarjeta result = TarjetaDAO.recuperarTarjeta(codigo);
        assertEquals(expResult.getCodigoRespuesta(), result.getCodigoRespuesta());
    }
}
```

UsuarioDAO

Verificar existencia de un usuario:

The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and JavaFXSistemaEstacionamientoJETS - Apache NetBeans IDE 13. The title bar shows the current project and file names: Start Page, JavaFXSistemaEstacionamientoJETS.java, RegistroDAOIT.java, TarjetaDAOIT.java, and UsuarioDAOIT.java.

The Navigator panel on the left lists various Java files under Source Packages and Test Packages. The Source tab of the main editor shows the code for `UsuarioDAOIT.java`, specifically the `testVerificarSession()` method. The code uses JUnit annotations (@After, @Test) and assertions to verify session functionality.

The Output panel at the bottom displays the test results: "Tests passed: 100.00 %" and "verificarSession". It also shows a message: "The test passed. (0.401 s)".

The status bar at the bottom right indicates the date and time: 12:51 a.m. 03/06/2023, and system information: INS Windows (CRLF).

10. Conclusión

Conclusión general:

El uso del modelo en cascada en el desarrollo del sistema de software presentó tanto ventajas como desventajas. Una de las desventajas principales fue la exclusión del cliente después de las primeras fases, donde se recopilan los requerimientos iniciales. Esto significa que no hubo retroalimentación por parte del cliente a lo largo del proceso, lo que podría haber llevado a desviaciones respecto a las expectativas del cliente y dificultades para realizar ajustes o mejoras según sus necesidades.

Por otro lado, una de las ventajas principales del modelo en cascada fue su enfoque lineal y secuencial en todas las fases del proyecto. Esto permitió seguir la planificación inicial de manera más estructurada y predecible. Cada fase se completó antes de pasar a la siguiente, lo que proporcionó una base sólida y una guía clara para el desarrollo del sistema.

En resumen, aunque el modelo en cascada brindó una secuencia lineal que facilitó la planificación inicial, la falta de retroalimentación del cliente fue una desventaja significativa que podría haber afectado la satisfacción del cliente y la adaptabilidad del sistema a sus necesidades cambiantes.

Sulem Martínez Aguilar:

En mi experiencia, emplear el modelo en cascada dificultó ciertos aspectos del desarrollo del sistema ya que, mientras íbamos avanzando en el diseño e

implementación nos dimos cuenta de que faltaba aclarar ciertos aspectos de partes previas, sin embargo, ya no podíamos regresarnos a corregirlas.

También, ocupar el modelo en cascada, tuvimos que tener mucho cuidado con la fase de requerimientos, y siento que fue en la que le dedicamos una buena parte del tiempo del desarrollo del sistema.

Y, finalmente, permitió que tuviéramos una secuencia de pasos estricta por lo que en cada fase teníamos muy claro cuándo tenía que finalizar la etapa actual para continuar con la siguiente, lo cual facilitó algunos aspectos.

Daniel Mongeote Tlachy:

Considero que la aplicación del modelo en cascada para futuros desarrollos de productos de software podría resultar en un producto final contraproducente, con la presencia de múltiples errores tanto por parte del cliente como de los desarrolladores. Además, es probable que existan importantes inconsistencias en relación con los requisitos planteados por el cliente.

El enfoque del modelo en cascada, al seguir una secuencia lineal y rigurosa de fases, podría limitar la flexibilidad y adaptabilidad necesarias para abordar cambios o ajustes en los requisitos a lo largo del proceso de desarrollo. La falta de interacción continua con el cliente, después de las primeras etapas de recopilación de requisitos, podría conducir a un producto final que no cumpla plenamente con las expectativas y necesidades del cliente.

Lo anterior lo experimentamos mi equipo y yo, ya que conforme avanzábamos individualmente con el desarrollo del documento, notábamos ciertos *huecos* en el planteamiento del proyecto, ya sea por falta de información o por mala interpretación de lo que se nos planteó en un primer lugar.

No obstante, si algo que puedo rescatar de la aplicación de este modelo, es la rígida y estricta secuencia de pasos asignados por el cronograma de actividades; este documento (junto con el PSP) fue, para mí, un “parteaguas” para el correcto manejo de mis actividades y la correcta gestión del tiempo que le debo dedicar a cada aspecto que deba realizar.

Jesús Jacob Montiel Salas:

En mi opinión, considero que, si bien no es el mejor modelo de desarrollo, el modelo en cascada pese a sus limitantes es un modelo que, como estudiantes, es con el que más estamos familiarizados y no sentí un gran cambio en algunas partes del desarrollo del proyecto, a diferencia de otros equipos los cuales pude notar que se encontraban un poco desorientados y perdidos al principio, así como poco acostumbrados a esas metodologías de trabajo.

Aun así, considero que es un modelo de trabajo bastante limitante y obsoleto para tiempos actuales donde en mi opinión es necesario mantener al cliente comprometido con el proyecto enseñándole avances del software mismo y no solo en forma de documentación, además de que este modelo no permite cambios ni

correcciones en las fases previas lo que limita enormemente la capacidad de adaptarnos a los cambios y sobre todo al propio error humano, complicando cada vez más las fases subsecuentes.

Pese a todo, considero que es una pena el no haber podido experimentar nuevos modelos de desarrollo para conocer su forma de trabajo más allá de una explicación en internet o en la propia clase, poder adaptarse a esa distinta forma de trabajo hubiera sido una experiencia interesante.

11. Formatos Personal Software Process (PSP)

PSP de Sulem Martínez Aguilar:

[Hoja de registro tiempos y defectos_MartínezAguilarSulem.xls](#)

PSP de Daniel Mongeote Tlachy:

[Hoja de registro tiempos y defectos - DanielMongeoteTlachy.xls](#)

PSP de Jesús Jacob Montiel Salas:

[Hoja de registro tiempos y defectos - Montiel Salas Jesús Jacob](#)

12. Bibliografía

IONOS Digital Guide. (2019, Marzo 21). *El Modelo en cascada: Desarrollo secuencial de software.* IONOS Digital Guide. Recuperado de: <https://www.ionos.mx/digitalguide/paginas-web/desarrollo-web/el-modelo-en-cascada/#:~:text=El%20modelo%20en%20cascada%20de,%2C%20implementaci%C3%B3n%20y%20mantenimiento>.

Blog Lucidchart. (2018, Septiembre 05). *Qué Te permite hacer (y qué no) La Metodología de Cascada para la Gestión de Proyectos.* Blog Lucidchart. Recuperado de: <https://www.lucidchart.com/blog/es/metodologia-gestion-proyectos-cascada>